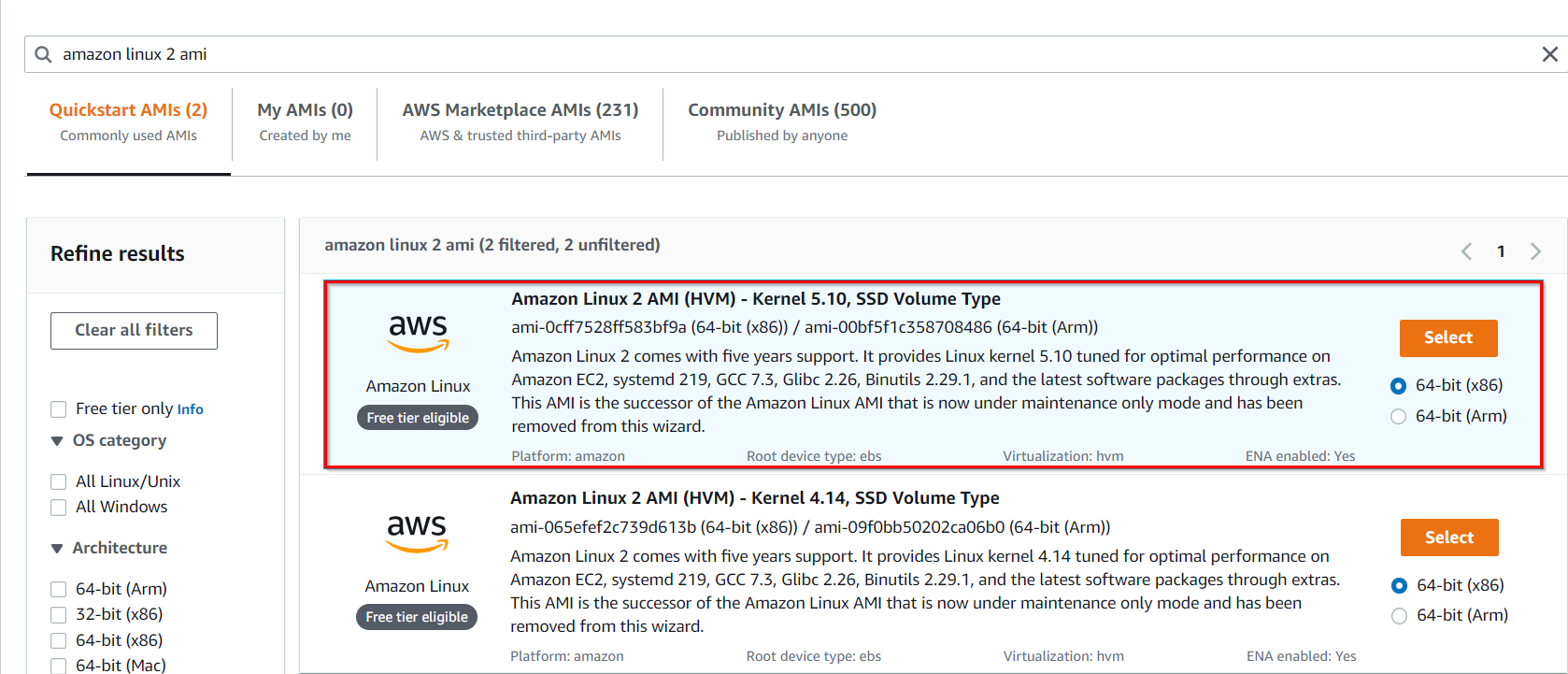
**Lab Steps**

**Task 1: Sign in to AWS Management Console**

1. Enter your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign in** button.
2. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1.**

**Task 2: Creating EC2 Instance**

1. Make sure you are in the **US East (N. Virginia)**Region.
2. Navigate to **EC2** by clicking on the **Services** menu in the top, then click on **EC2** in the **Compute** section.
3. Click on **Instances** from the left side bar and then click on **Launch Instances**
4. While launching the instance, you will notice that one instance named **labs\_instance** will already be present there. We will use that instance for **CLI** for creating ALB.
5. Name : Enter ***Instance1***
6. **For Amazon Machine Image (AMI):** Search for **Amazon Linux 2 AMI** in the search box and click on the **select** button.

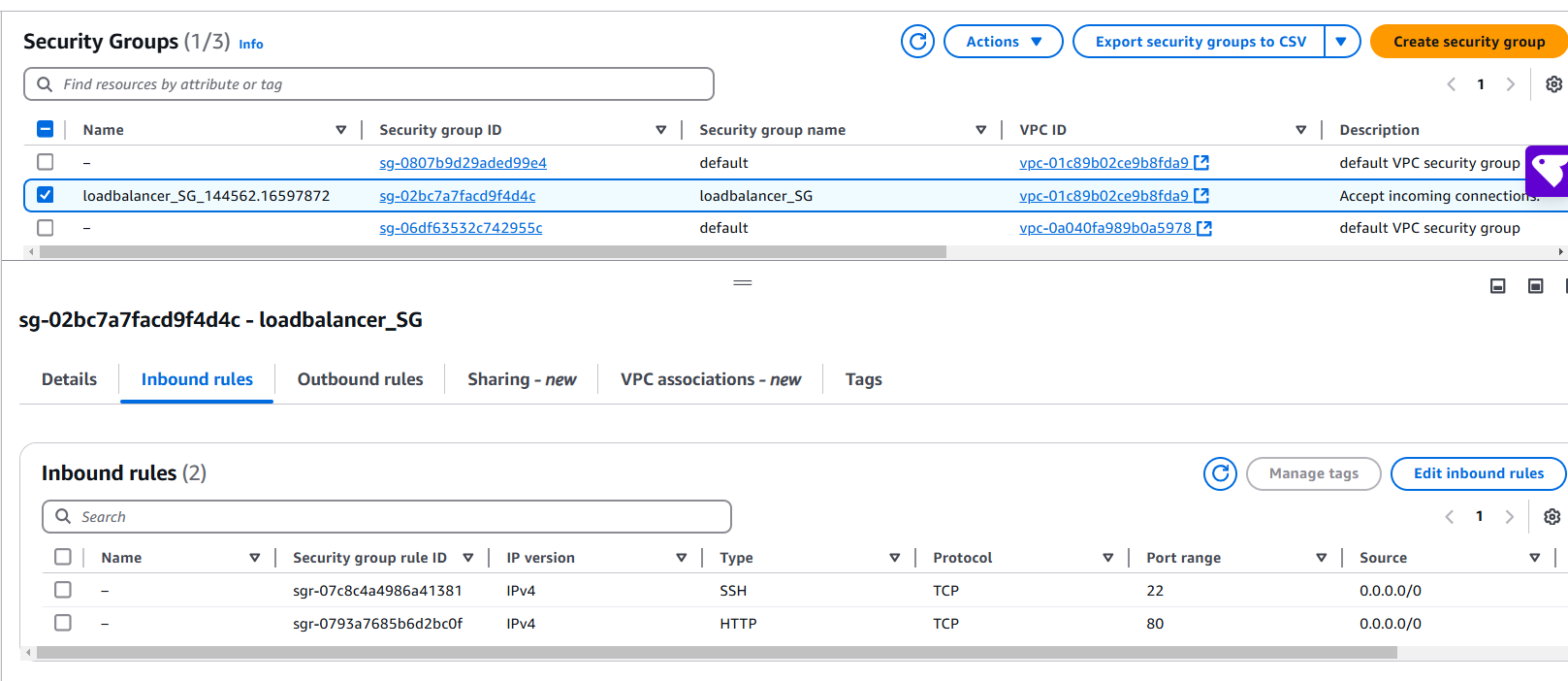


1. **For Instance Type:** select ***t2.micro***

******

1. **For Key pair:** Select **Create a new key pair** Button
   * Key pair name: ***MyEC2-key***
   * Key pair type: **RSA**
   * Private key file format: **.pem**
2. Select **Create key pair** Button.
3. In Network Settings Click on **Edit**:

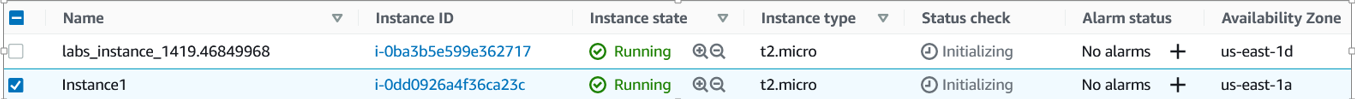
* VPC : Select **Default VPC** from the available VPCs
* Subnet: Select **Default in us-east-1a** (you can choose any one availability zones from the list of subnets).
* Auto-assign public IP: **Enable**
* Click on **Select an Existing Security Group**
* Select the security group name **loadbalancer\_SG** (which will have inbound port 22 and 80 open for all traffic, please check below screenshot)



1. Click on**Advanced Details** when you scroll down**.**
2. In **User data** section, enter the following script to make the server a **web server.**

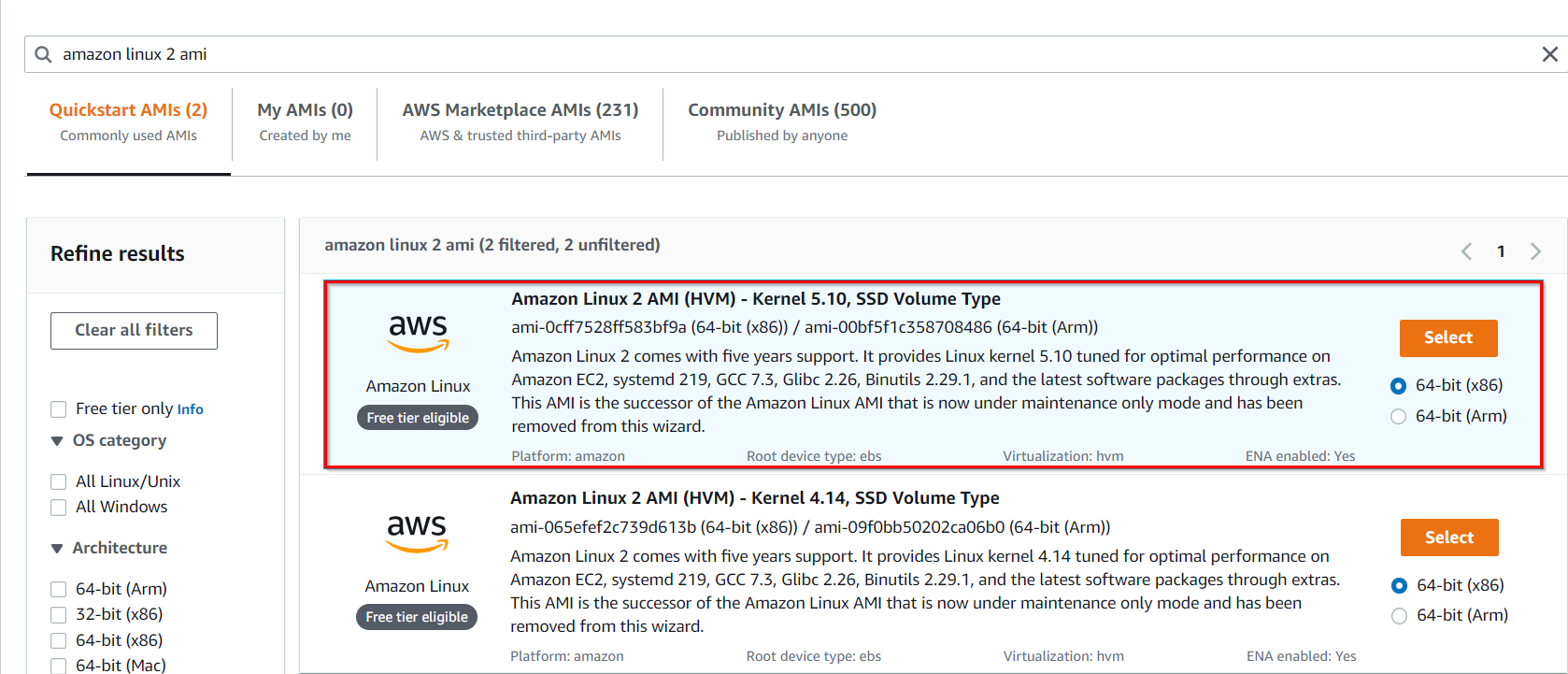
|  |
| --- |
| #!/bin/bash  sudo su  yum update -y  yum install httpd -y  touch /var/www/html/index.html  echo “REQUEST SERVED FROM INSTANCE1” >> /var/www/html/index.html  chmod 777 /var/www/html/index.html  mkdir -p /var/www/html/work  touch /var/www/html/work/test.html  echo “REQUEST SERVED FROM WORK PATH OF INSTANCE1” >> /var/www/html/work/test.html  mkdir -p /var/www/html/images  touch /var/www/html/images/test.html  echo “REQUEST SERVED FROM IMAGES PATH OF INSTANCE1” >> /var/www/html/images/test.html  systemctl start httpd  systemctl enable httpd |

1. Click on **Launch Instances**.
2. After 1-2 minutes, the **Instance State** will change to **running** as shown below:



**Task 3: Creating another EC2 Instance**

1. Make sure you are in the **US East (N. Virginia)**Region.
2. Click on **Launch Instances**
3. Name : Enter ***Instance2***
4. **For Amazon Machine Image (AMI):** Search for **Amazon Linux 2 AMI** in the search box and click on the **select** button.



1. **For Instance Type:** select *t2.micro*

****

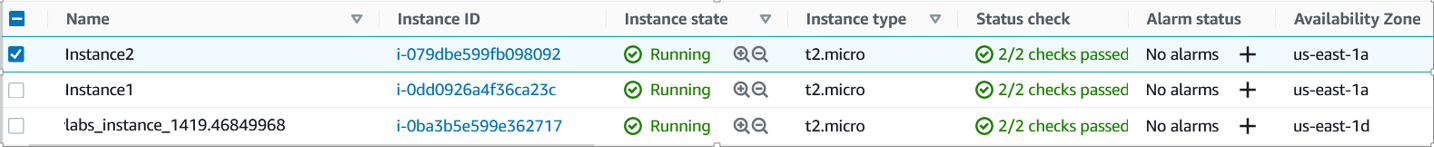
1. **For Key pair:** Select **Existing Key pair**
2. In Network Settings Click on **Edit**:

* VPC : Select **Default VPC** from the available VPCs
* Subnet: Select **Default in us-east-1b** (you can choose any one availability zones from the list of subnets).
* Auto-assign public IP: **Enable**
* Click on **Select an Existing Security Group**
* Select the security group name **loadbalancer\_SG** (which will have inbound port 22 and 80 open for all traffic)

1. Click on**Adanved Details** when you scroll down**.**
2. In **User data** section, enter the following script to make the server a **web server.**

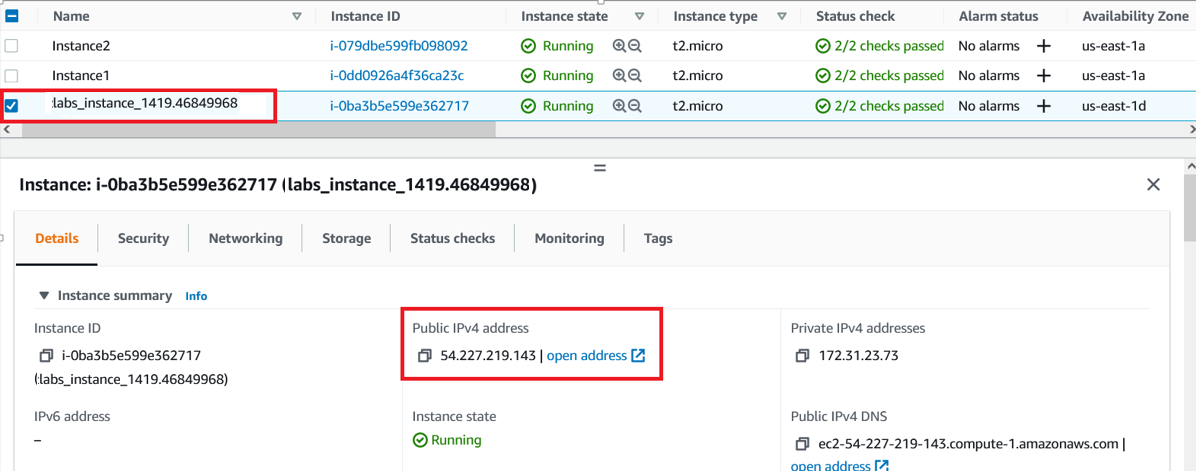
|  |
| --- |
| #!/bin/bash  sudo su  yum update -y  yum install httpd -y  touch /var/www/html/index.html  echo “REQUEST SERVED FROM INSTANCE2” >> /var/www/html/index.html  chmod 777 /var/www/html/index.html  mkdir -p /var/www/html/work  touch /var/www/html/work/test.html  echo “REQUEST SERVED FROM WORK PATH OF INSTANCE2” >> /var/www/html/work/test.html  chmod 777 /var/www/html/work/test.html  mkdir -p /var/www/html/images  touch /var/www/html/images/test.html  echo “REQUEST SERVED FROM IMAGES PATH OF INSTANCE2” >> /var/www/html/images/test.html  chmod 777 /var/www/html/images/test.html  systemctl start httpd  systemctl enable httpd |

1. Keep Rest thing Default and Click on **Launch Instance** Button.
2. Select **View all Instances** to View Instance you Created
3. After 1-2 Mins **Instance State** will become **running** as shown below



**Task 4: Creating an Application Load Balancer in AWS CLI**

1. Make sure you are in the **US East (N. Virginia)**Region.
2. In the EC2 dashboard, apart from the 2 instances you have launched manually, there will be another instance named **labs\_instance.** Select the instance and copy its **Public IPv4** **Address**.



1. **SSH**into that **labs\_instance.**
2. For **Windows users** , **SSH** into the **labs\_instance** by downloading **putty** from this link [**https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html**](https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html) and then enter the following command in the**Host Name (or IP address)** section:
3. Once you **SSH** into the **labs\_instance**, configure your server by executing the below command to eliminate adding the region in each command:

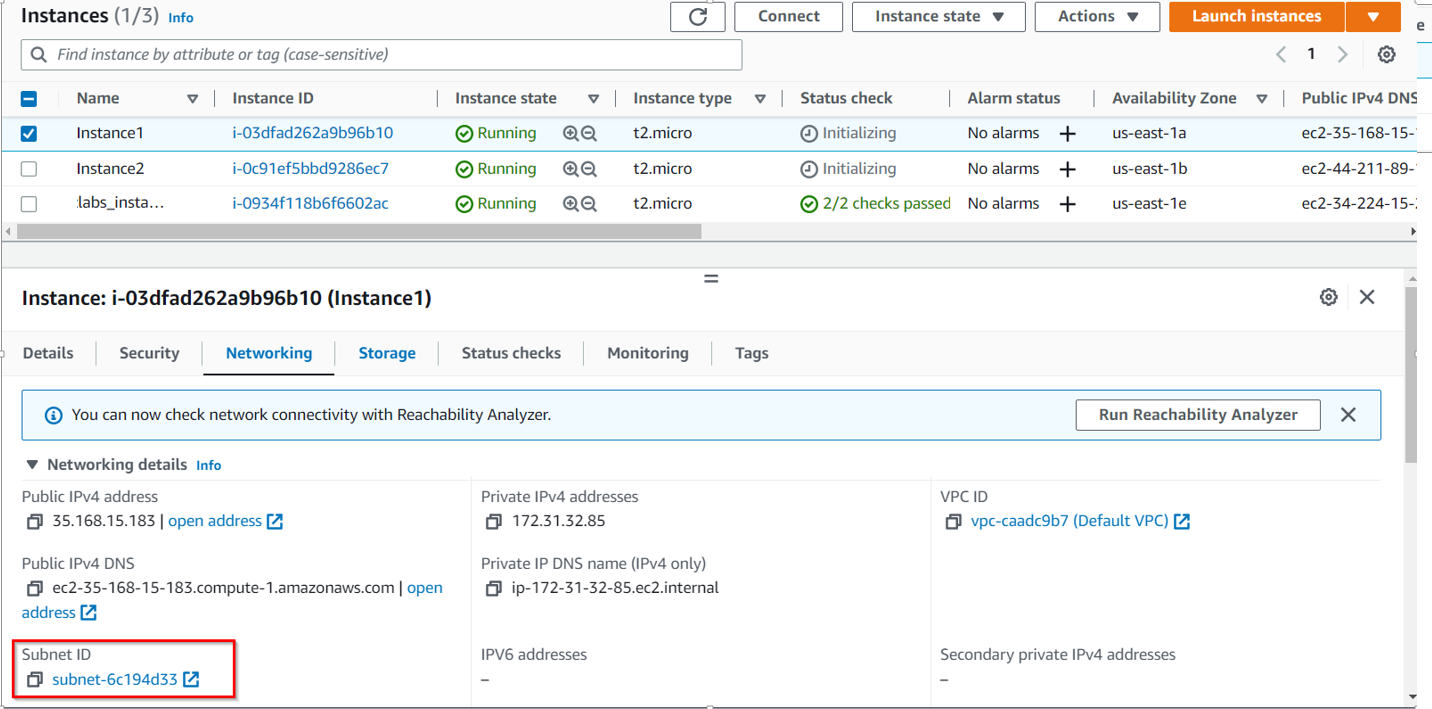
aws configure

1. Press enter for both the AWS Access key and AWS Secret key and **enter us-east-1 in the Default Region field**.

* AWS Access key ID  : Press **Enter**
* AWS  Secret Key  :  Press **Enter**
* Default region name  : Enter ***us-east-1***
* Default output format  :  Press **Enter**

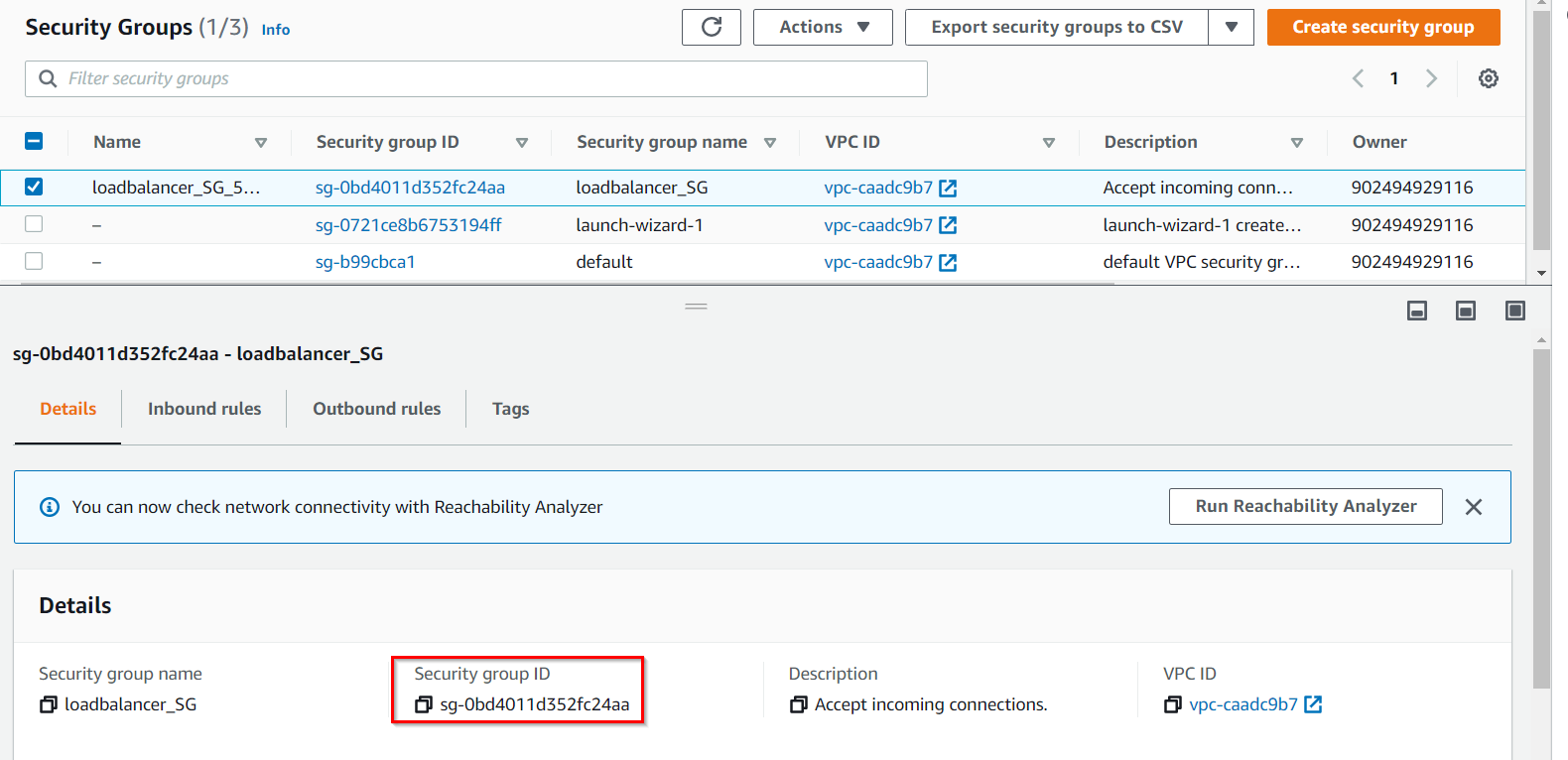
**Task 5: Creating a Load Balancer**

1. Make sure you are in the **US East (N. Virginia)**Region.
2. Go to the EC2 dashboard and select the instance named **Instance1.** Copy its **Subnet ID** from the description page to a text file.



1. Similarly, select **Instance2** and copy its **Subnet ID** and store it into a text file.

1. Copy the Security Group ID of the **Security Group** named **loadbalancer\_SG**and store it into a text file.

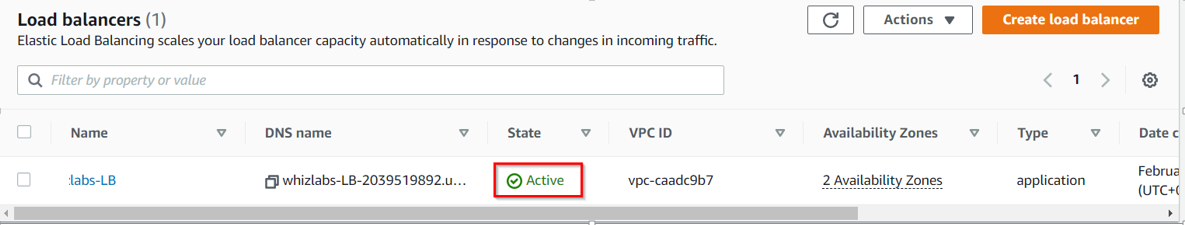


1. In the terminal, use the **create-load-balancer** command to create a **load balancer**named **labs-LB** (which includes two subnets in different availability zones, specifically **us-east-1a** and **us-east-1b**). Enter the following command into the terminal as shown below:

aws elbv2 create-load-balancer --name labs-LB --subnets <Subnet Id of Instance 1> <Subnet Id of Instance 2> --security-groups <Security Group of loadbalancer\_SG>

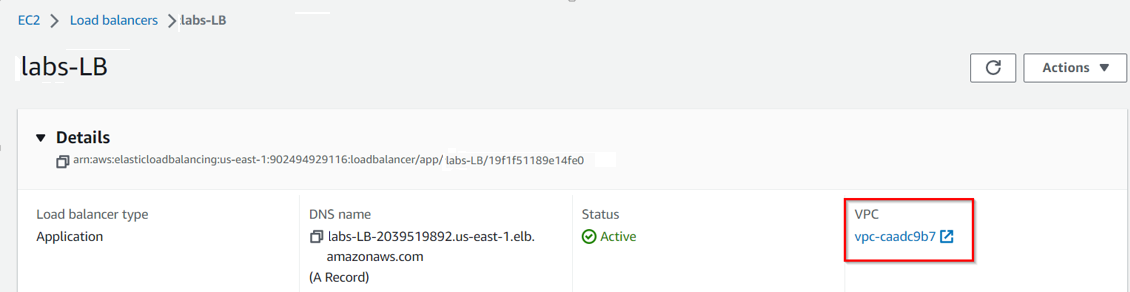
* **Example: aws elbv2 create-load-balancer --name labs-LB --subnets subnet-09dcfc9d353ee9c5c subnet-067e6abb9582e5805 --security-groups sg-0ef883fe5d7d2575c**
* **Note: Replace the Subnet Id of Instance 1, Subnet Id of Instance 2 and Security Group of loadbalancer\_SG with yours.**
* Copy the **Load Balancer’s ARN**

1. Go to the **EC2 dashboard**, click onthe **Load Balancer** (on the left-hand side menu) and see whether the Load Balancer is created. Make sure that the **state** of the Load Balancer is **Active.**Wait **5 minutes** for the state to change from **Provisioning to Active**.



**Task 6: Creating 2 Target Groups**

1. Make sure you are in the **US East (N. Virginia)**Region.
2. Go to **Services**and click on**EC2**. Select Load Balancers from the left-hand side menu.
3. Select **labs-LB** and copy the **VPC-ID** to a text file.



1. Use the **create-target-group** command to create a target group named **TG1,** specify the load balancer's VPC ID.

aws elbv2 create-target-group --name TG1 --protocol HTTP --port 80 --vpc-id <Load balancer VPC Id>

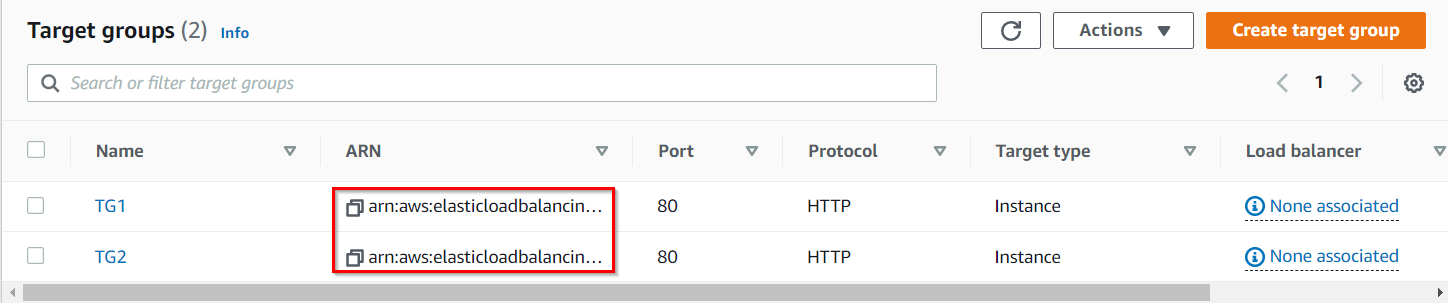
* **Example: aws elbv2 create-target-group --name TG1 --protocol HTTP --port 80 --vpc-id vpc-0ec3696a4c41fed32**
* **Note: Replace the Load balancer VPC Id with your value.**

1. Similarly, use the **create-target-group** command to create another target group named TG2. Specify the same VPC ID as above.

aws elbv2 create-target-group --name TG2 --protocol HTTP --port 80 --vpc-id <Load balancer VPC Id>

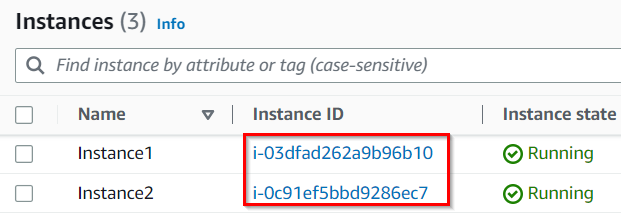
* **Example: aws elbv2 create-target-group --name TG2 --protocol HTTP --port 80 --vpc-id vpc-0ec3696a4c41fed32**
* **Note: Replace the Load balancer VPC Id with your value.**

1. From the AWS console, go to the **EC2 dashboard**, click on **Target Groups**(on the left-hand side menu). Once the Target Group is created, copy and paste the **Target Group** **ARNs** of the target groups **TG1 and TG2 (separately)** from the description page to a text file.



**Task 7: Register the Targets with their Respective Target groups**

1. Go to **Services,** click on **EC2** and select the instances one by one and copy the **instance ids** of **Instance1** and **Instance2**from the Description Tab to a **text file.**



1. Use **register-targets** command to register **Instance1** with the **Target Group TG1.**

aws elbv2 register-targets --target-group-arn <TG1 ARN> --targets Id=<Instance1 Id>

* **Example: aws elbv2 register-targets --target-group-arn arn:aws:elasticloadbalancing:us-east-1:757712384777:targetgroup/TG1/f4619175d2877cd9 --targets Id=i-0d9d44b763709b07d**
* **Note: Replace the TG1 ARN and INSTANCE1 ID with your values for each.**

1. Similarly, use the **register-targets** command to register **Instance2** with the **Target Group TG2.**

aws elbv2 register-targets --target-group-arn <TG2 ARN> --targets Id=<Instance2 Id>

* **Example: aws elbv2 register-targets --target-group-arn arn:aws:elasticloadbalancing:us-east-1:757712384777:targetgroup/TG2/1dd2532226541d8c --targets Id=i-016edc45d166c9123**
* **Note: Replace the TG2 ARN and INSTANCE2 ID with your values for each.**

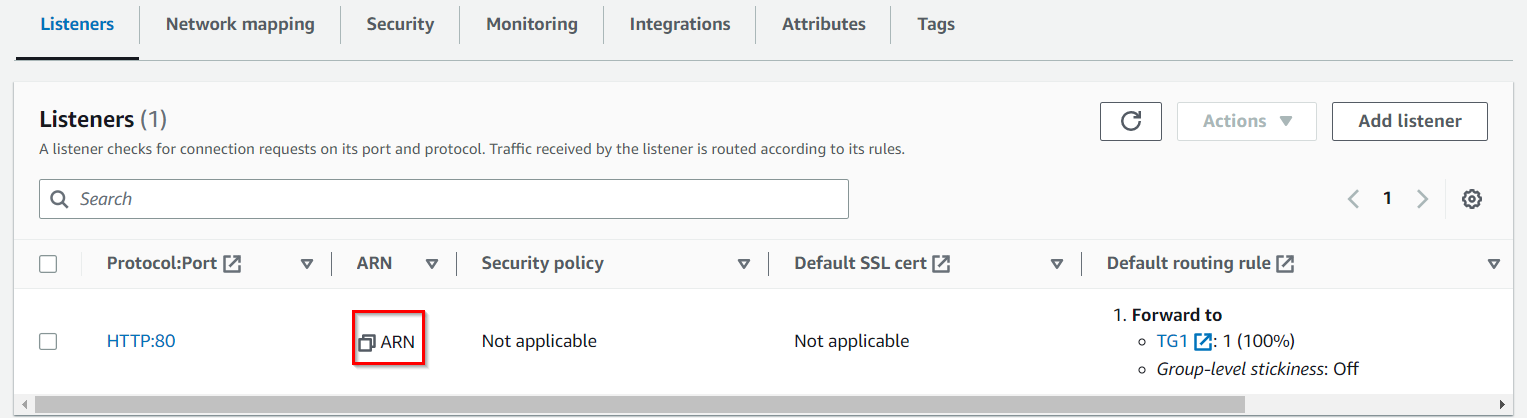
**Task 8: Creating Listener Default Rules**

1. Use the **create-listener** command to create a **listener default rule** to **forward the request to Target group TG1.**

aws elbv2 create-listener --load-balancer-arn <Load Balancer ARN> --protocol HTTP --port 80 --default-actions Type=forward,TargetGroupArn=<TG1 ARN>

* **Example: aws elbv2 create-listener --load-balancer-arn arn:aws:elasticloadbalancing:us-east-1:757712384777:loadbalancer/app/whizlabs-LB/c70ba276f3e59d69 --protocol HTTP --port 80 --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-east-1:757712384777:targetgroup/TG1/f4619175d2877cd9**
* **Note: Replace the Load Balancer ARN and TG1 ARN with your values for each.**

1. Once you have created the listener default rule, navigate to the**EC2 Dashboard** and click on **Load Balancers**. Copy the **Listeners ARN** from the **Listeners Tab** by hovering over the **arn** and save it a text file.



**Task 9: Creating Listeners for other rules**

1. Use the **create-listener** command along with the **Load Balancer listeners ARN** to create a **listener rule1** to forward the **request to TG1** if its **URL** has **images** in its path.

aws elbv2 create-rule --listener-arn <LB Listeners ARN> --priority 10 --conditions Field=path-pattern,Values='/images/\*' --actions Type=forward,TargetGroupArn=<TG1 ARN>

* **Example: aws elbv2 create-rule --listener-arn arn:aws:elasticloadbalancing:us-east-1:757712384777:listener/app/whizlabs-LB/c70ba276f3e59d69/12d71a05863d10c1 --priority 10 --conditions Field=path-pattern,Values='/images/\*' --actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-east-1:757712384777:targetgroup/TG1/f4619175d2877cd9**
* **Note: Replace the LB Listeners ARN and TG1 ARN  with your values for each.**

1. Use the **Create-listener** command along with the **Load Balancer listener ARN**to create a **listener rule** to forward the **request to TG2** if its **URL** has **work** in its path. Also kindly make sure the priority for this rule is different from the previous rule we created.

aws elbv2 create-rule --listener-arn <LB Listeners ARN> --priority 5 --conditions Field=path-pattern,Values='/work/\*' --actions Type=forward,TargetGroupArn=<TG2 ARN>

* **Example: aws elbv2 create-rule --listener-arn arn:aws:elasticloadbalancing:us-east-1:757712384777:listener/app/whizlabs-LB/c70ba276f3e59d69/12d71a05863d10c1 --priority 5 --conditions Field=path-pattern,Values='/work/\*' --actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-east-1:757712384777:targetgroup/TG2/1dd2532226541d8c**
* **Note: Replace the LB Listeners ARN and TG2 ARN  with your values for each.**

**Task 10: Verifying the health of the Target Groups**

1. Use the **describe-target-health** command to verify the health status of the **Target Group TG1.**

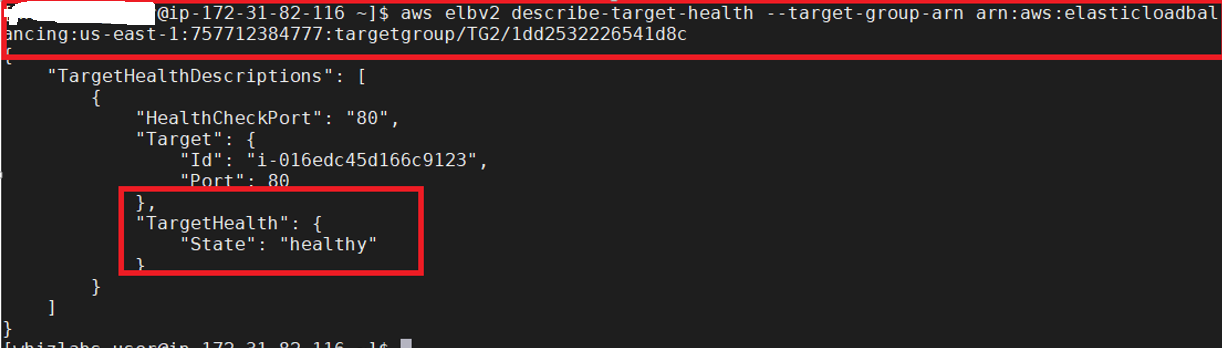
aws elbv2 describe-target-health --target-group-arn <TG1 ARN>

* **Example: aws elbv2 describe-target-health --target-group-arn arn:aws:elasticloadbalancing:us-east-1:757712384777:targetgroup/TG1/f4619175d2877cd9**
* **NOTE: Replace the TG1 ARN with your value.**

1. Similarly, use the **describe-target-health** command to verify the health status of the **Target Group TG2.**

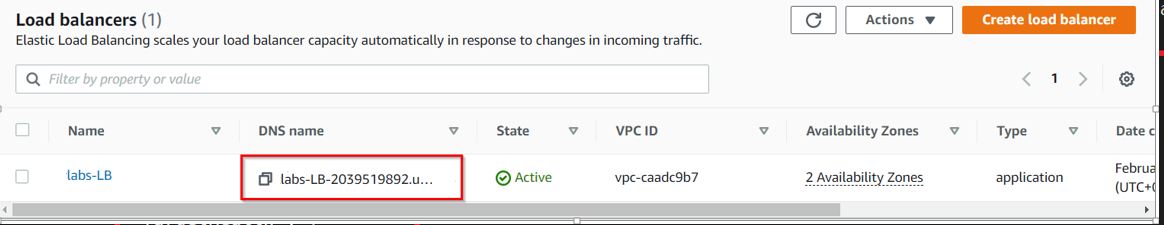
aws elbv2 describe-target-health --target-group-arn <TG2 ARN>

* **Example: aws elbv2 describe-target-health --target-group-arn arn:aws:elasticloadbalancing:us-east-1:757712384777:targetgroup/TG2/1dd2532226541d8c**
* **NOTE: Replace the TG2 ARN with your value.**

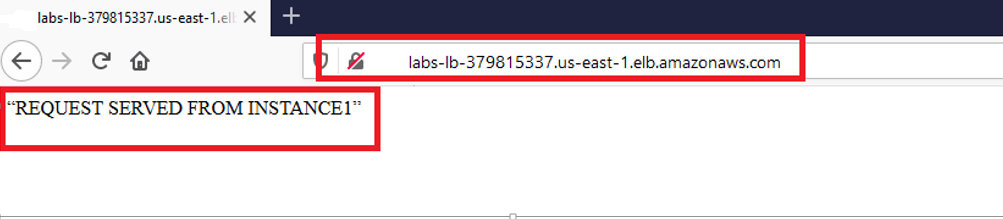


**Task 11: Verifying the Load balancer rules by accessing the DNS**

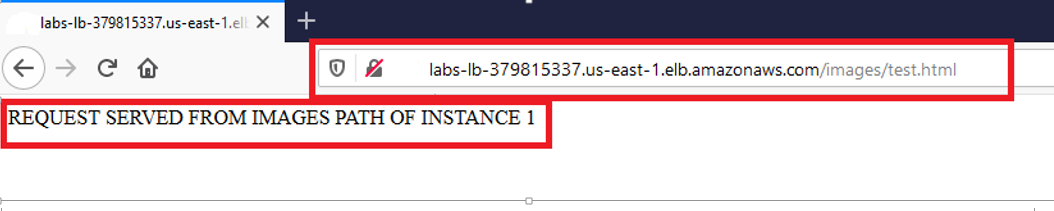
1. Make sure you are in the **US East (N. Virginia)**Region.
2. Go to **Services** and select **Load Balancer**from the **EC2 dashboard**. Select **whizlabs-LB** and **copy the Load Balancer DNS Name.**



1. Try to access the**DNS Name** from the **Browser** and verify that you can able to get the output of **Instance1's html page** ( i.e it has to serve the request from TG1 because the default listener rule should route the traffic to TG1).

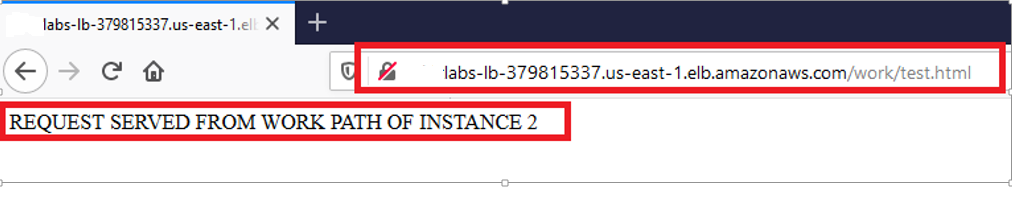


1. Now try to access the **DNS Name along** with the ***/images/test.html*** in its path (as given below) and verify that it has to serve the **request from Instance1** (i.e if the DNS URL path contains the name "images" then it has to route the request to the target group TG1).

* **Note: Use your Load Balancer DNS Name and append */images/test.html* at the end.**  
   

1. Now try to access the **DNS Name along** with the ***/work/test.html*** in its path (as given below) and verify that it has to serve the **request from instance2** (i.e if the DNS URL path contains the name "work" then it has to route the request to the target group TG2).

* **Note: Use your Load Balancer DNS Name and append */work/test.html* at the end.**



**Do You Know ?**

When using the AWS Command Line Interface (CLI) to create an Application Load Balancer, you can specify multiple subnets in different Availability Zones (AZs) as part of the creation process. This enables you to distribute the load balancer's traffic across multiple AZs for improved fault tolerance and high availability.

**Task 12: Delete AWS Resources**

**Deleting Load Balancer and Target Groups**

1. To delete Application Load Balancer, copy the ARN and replace it in the command below:

aws elbv2 delete-load-balancer --load-balancer-arn <Load Balancer ARN>

* + **Example: aws elbv2 delete-load-balancer --load-balancer-arn arn:aws:elasticloadbalancing:us-east-1:416177121910:loadbalancer/app/whizlabs-LB/5d9d010d070a48e1**

1. To delete first target group i.e. TG1, copy the ARN and replace it in the command below:

aws elbv2 delete-target-group --target-group-arn <TG1 ARN>

* + **Example: aws elbv2 delete-target-group --target-group-arn arn:aws:elasticloadbalancing:us-east-1:416177121910:targetgroup/TG1/9e2d879c2b029051**

1. To delete second target group i.e. TG copy the ARN and replace it in the command below:

aws elbv2 delete-target-group --target-group-arn <TG2 ARN>

* + **Example: aws elbv2 delete-target-group --target-group-arn arn:aws:elasticloadbalancing:us-east-1:416177121910:targetgroup/TG2/04bf266a2aa4c7b9**

**Deleting EC2 Instances**

1. To terminate the EC2 Instances, you can terminate all at once:

aws ec2 terminate-instances --instance-ids <instance id of Instance1> <instance id of Instance2> <instance id of labs\_instance>

* **Example: aws ec2 terminate-instances --instance-ids i-0e3190473a41c6e64 i-05ff676f71f0ed3bf i-08a139e67e9eebaf3**

**Completion and Conclusion**

1. You have successfully created 2 EC2 instances in the default VPC.
2. You have successfully SSH into the already-available labs\_instance.
3. You have successfully configured the labs\_instance in us-east-1 using AWS CLI commands.
4. You have successfully created an Application Load Balancer using AWS CLI command.
5. You have successfully created 2 Target Groups via AWS CLI command.
6. You have successfully registered EC2 instances with their Target Groups via AWS CLI command.
7. You have successfully created a default listener rule via AWS CLI command.
8. You have successfully created 2 rules to route the traffic to a separate Target Group based on a path using AWS CLI command.
9. You have successfully verified the health of the targets using AWS CLI command.
10. You have successfully accessed the DNS Name of the load balancer from a browser and verified that the routing is done according to the specified rules.