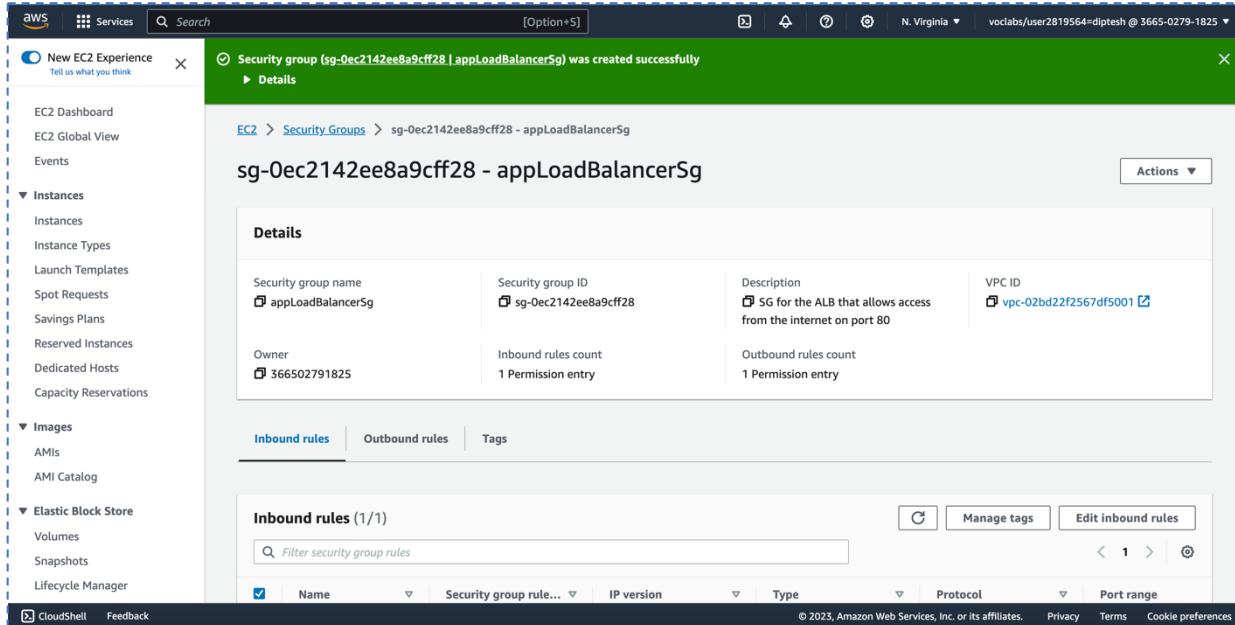


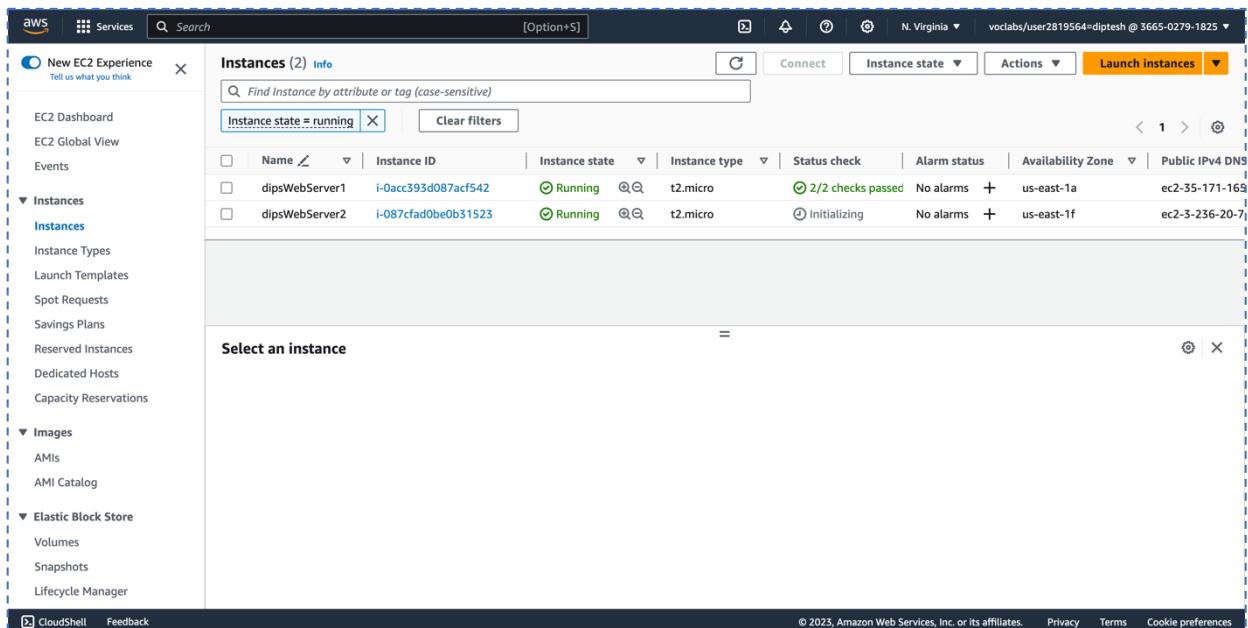
## ASSIGNMENT - 4

- **Task 1: Run 2 servers behind ALB**



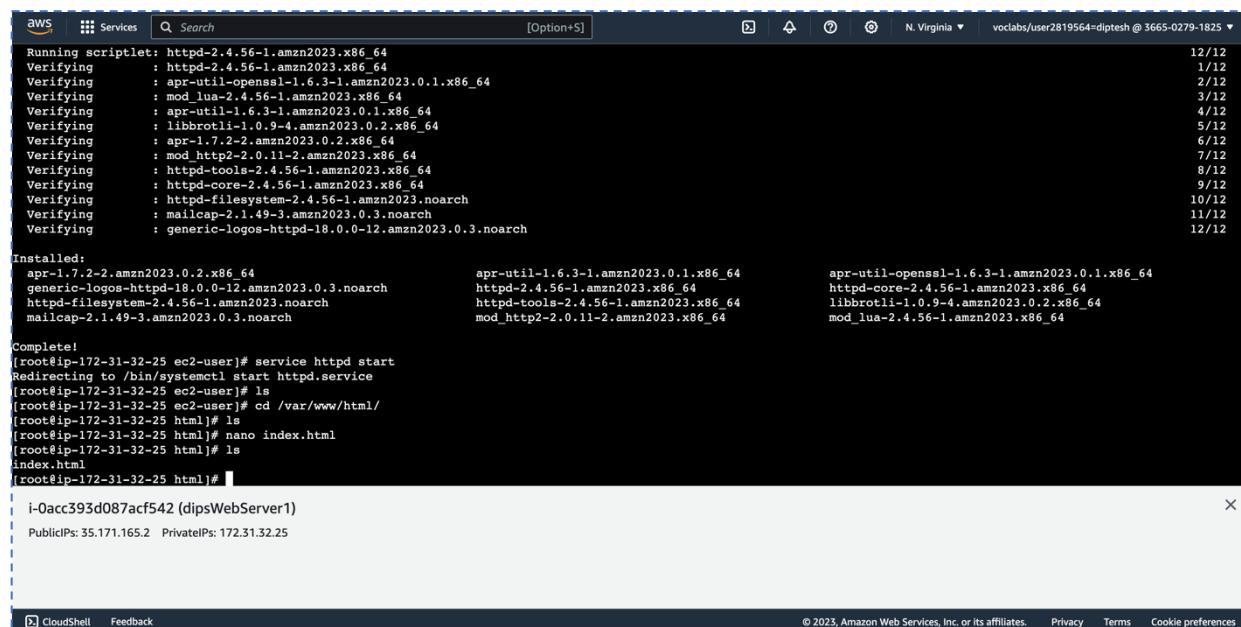
The screenshot shows the AWS EC2 Security Groups page. A green banner at the top indicates that a security group was created successfully. The main section displays the details of the security group 'sg-Oec2142ee8a9cff28 - appLoadBalancerSg'. It includes fields for Security group name (appLoadBalancerSg), Security group ID (sg-Oec2142ee8a9cff28), Description (SG for the ALB that allows access from the internet on port 80), VPC ID (vpc-02bd22f2567df5001), Owner (366502791825), Inbound rules count (1 Permission entry), and Outbound rules count (1 Permission entry). Below this, there are tabs for Inbound rules, Outbound rules, and Tags, with the Inbound rules tab selected. An 'Inbound rules (1/1)' table is shown with one rule entry.

Image -1: Security group created for Application Load Balance.



The screenshot shows the AWS EC2 Instances page. The left sidebar lists various EC2-related services. The main area shows a table of instances. There are two entries: 'dipsWebServer1' (Instance ID: i-0acc393d087acf542) and 'dipsWebServer2' (Instance ID: i-087cfad0be0b31523). Both instances are listed as 'Running' with 't2.micro' instance type, '2/2 checks passed' status, and no alarms. They are located in the 'us-east-1a' availability zone and have public IPv4 DNS names 'ec2-35-171-163' and 'ec2-3-236-20-7'. A modal window titled 'Select an instance' is open at the bottom.

Image -2: Two instances of EC2 created.



```

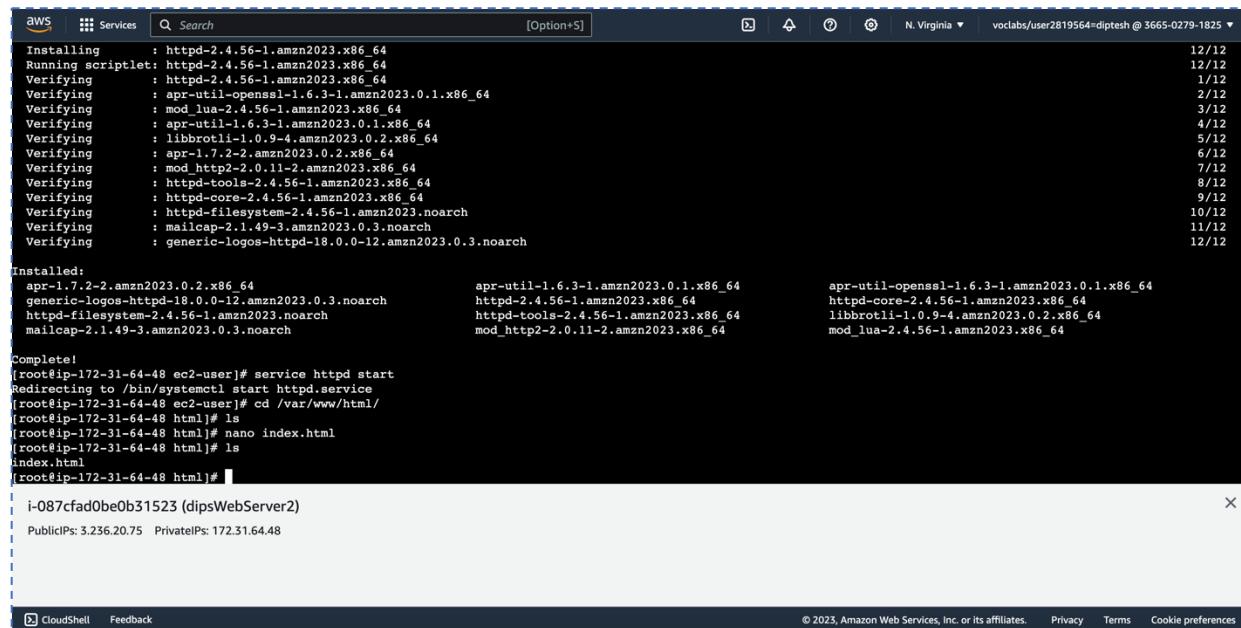
AWS Services Search [Option+S] N. Virginia vocabs/user2819564=diptesh@3665-0279-1825 ▾
Running scriptlets: httpd-2.4.56-1.amzn2023.x86_64 12/12
Verifying : httpd-2.4.56-1.amzn2023.x86_64 1/12
Verifying : apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64 2/12
Verifying : mod_lua-2.4.56-1.amzn2023.x86_64 3/12
Verifying : apr-util-1.6.3-1.amzn2023.0.1.x86_64 4/12
Verifying : libbrotli-1.0.9-4.amzn2023.0.2.x86_64 5/12
Verifying : apr-1.7.2-2.amzn2023.0.2.x86_64 6/12
Verifying : mod_http2-2.0.11-2.amzn2023.x86_64 7/12
Verifying : httpd-tools-2.4.56-1.amzn2023.x86_64 8/12
Verifying : httpd-core-2.4.56-1.amzn2023.x86_64 9/12
Verifying : httpd-filesystem-2.4.56-1.amzn2023.noarch 10/12
Verifying : mailcap-2.1.49-3.amzn2023.0.3.noarch 11/12
Verifying : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch 12/12

Installed:
apr-1.7.2-2.amzn2023.0.2.x86_64
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
httpd-filesystem-2.4.56-1.amzn2023.noarch
mod_http2-2.1.49-3.amzn2023.0.3.noarch
mod_lua-2.4.56-1.amzn2023.0.3.noarch
apr-util-1.6.3-1.amzn2023.0.1.x86_64
httpd-2.4.56-1.amzn2023.x86_64
httpd-tools-2.4.56-1.amzn2023.x86_64
mod_http2-2.0.11-2.amzn2023.x86_64
libbrotli-1.0.9-4.amzn2023.0.2.x86_64
mod_lua-2.4.56-1.amzn2023.x86_64

Complete!
[root@ip-172-31-32-25 ec2-user]# service httpd start
Redirecting to /bin/systemctl start httpd.service
[root@ip-172-31-32-25 ec2-user]# ls
[root@ip-172-31-32-25 ec2-user]# cd /var/www/html/
[root@ip-172-31-32-25 html]# ls
[root@ip-172-31-32-25 html]# nano index.html
[root@ip-172-31-32-25 html]# ls
index.html
[root@ip-172-31-32-25 html]# 

i-Qacc393d087acf542 (dipsWebServer1)
PublicIPs: 35.171.165.2 PrivateIPs: 172.31.32.25

```

Image -3: Configured web server 1 in first EC2 instance.


```

AWS Services Search [Option+S] N. Virginia vocabs/user2819564=diptesh@3665-0279-1825 ▾
Installing : httpd-2.4.56-1.amzn2023.x86_64 12/12
Running scriptlets: httpd-2.4.56-1.amzn2023.x86_64 1/12
Verifying : httpd-2.4.56-1.amzn2023.x86_64 2/12
Verifying : apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64 3/12
Verifying : mod_lua-2.4.56-1.amzn2023.x86_64 4/12
Verifying : apr-util-1.6.3-1.amzn2023.0.1.x86_64 5/12
Verifying : libbrotli-1.0.9-4.amzn2023.0.2.x86_64 6/12
Verifying : apr-1.7.2-2.amzn2023.0.2.x86_64 7/12
Verifying : mod_http2-2.0.11-2.amzn2023.x86_64 8/12
Verifying : httpd-tools-2.4.56-1.amzn2023.x86_64 9/12
Verifying : httpd-core-2.4.56-1.amzn2023.x86_64 10/12
Verifying : httpd-filesystem-2.4.56-1.amzn2023.noarch 11/12
Verifying : mailcap-2.1.49-3.amzn2023.0.3.noarch 12/12
Verifying : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch

Installed:
apr-1.7.2-2.amzn2023.0.2.x86_64
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
httpd-filesystem-2.4.56-1.amzn2023.noarch
mod_http2-2.1.49-3.amzn2023.0.3.noarch
mod_lua-2.4.56-1.amzn2023.0.3.noarch
apr-util-1.6.3-1.amzn2023.0.1.x86_64
httpd-2.4.56-1.amzn2023.x86_64
httpd-tools-2.4.56-1.amzn2023.x86_64
mod_http2-2.0.11-2.amzn2023.x86_64
libbrotli-1.0.9-4.amzn2023.0.2.x86_64
mod_lua-2.4.56-1.amzn2023.x86_64

Complete!
[root@ip-172-31-64-48 ec2-user]# service httpd start
Redirecting to /bin/systemctl start httpd.service
[root@ip-172-31-64-48 ec2-user]# cd /var/www/html/
[root@ip-172-31-64-48 html]# ls
[root@ip-172-31-64-48 html]# nano index.html
[root@ip-172-31-64-48 html]# ls
index.html
[root@ip-172-31-64-48 html]# 

i-087cfad0be0b31523 (dipsWebServer2)
PublicIPs: 3.236.20.75 PrivateIPs: 172.31.64.48

```

Image -4: Configured web server 2 in another EC2 instance.

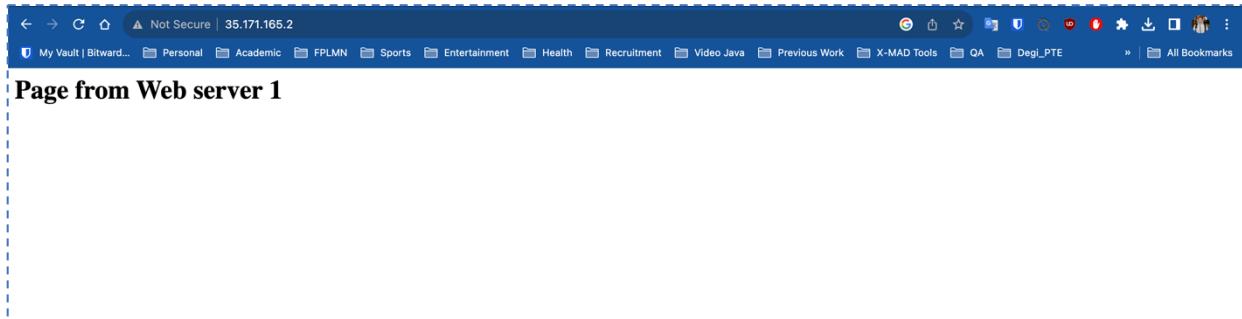


Image -5: Browsing application in web server 1.

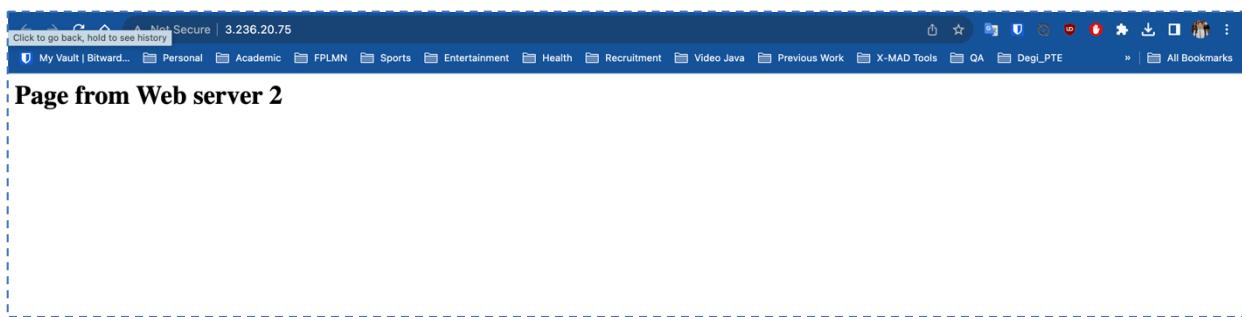


Image -6: Browsing application in web server 2.

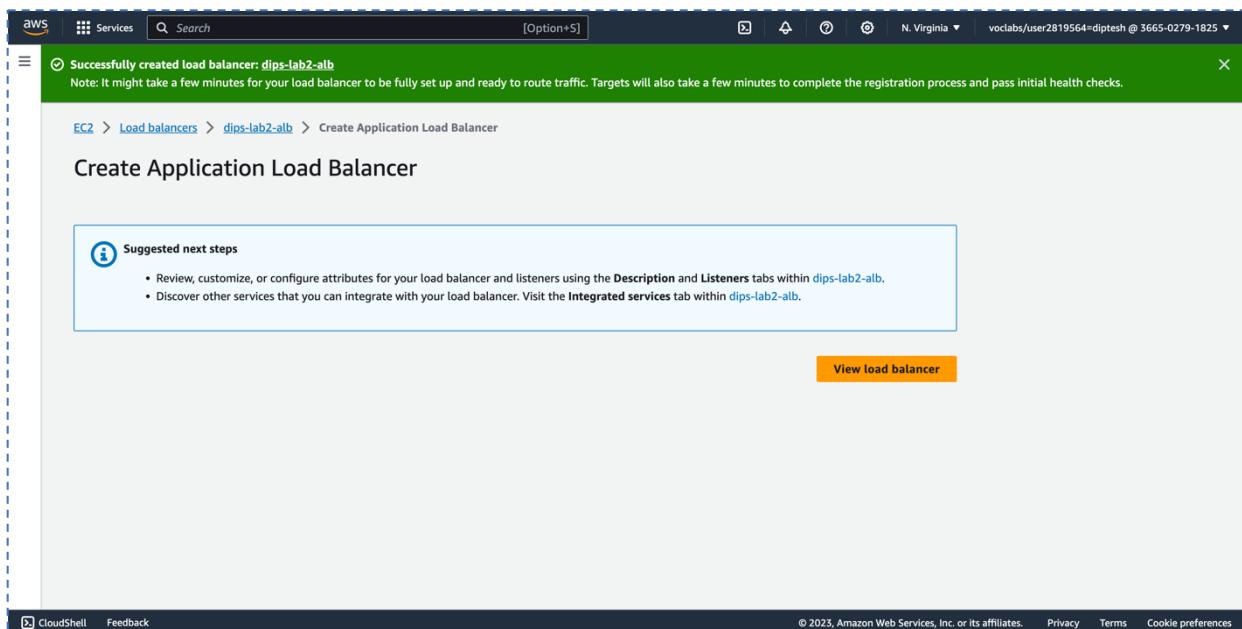


Image -7: Created and configured new ALB.

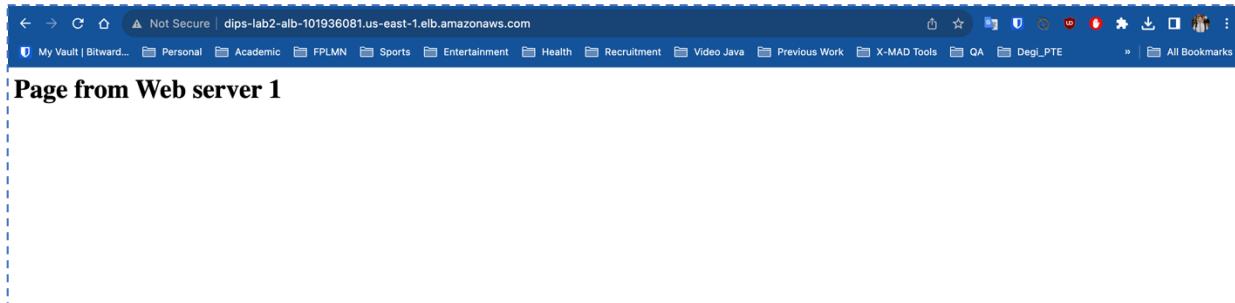


Image -8: Browsing application in web server 1 via ALB.

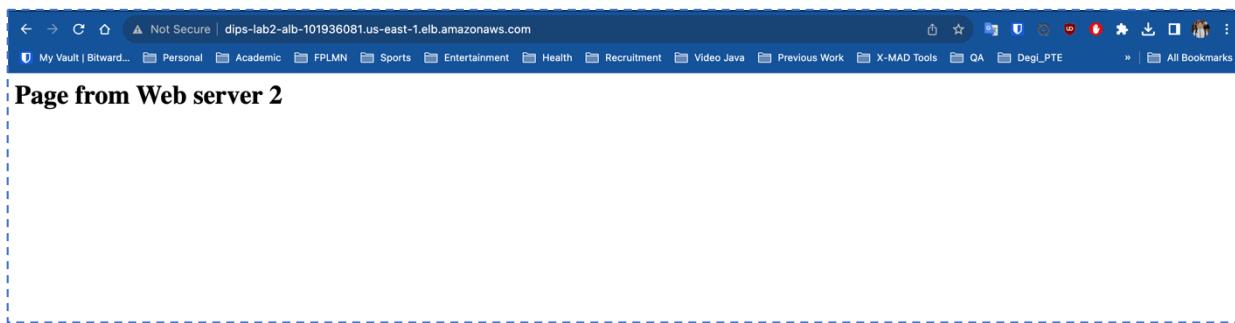


Image -9: Browsing application in web server 2 via ALB.

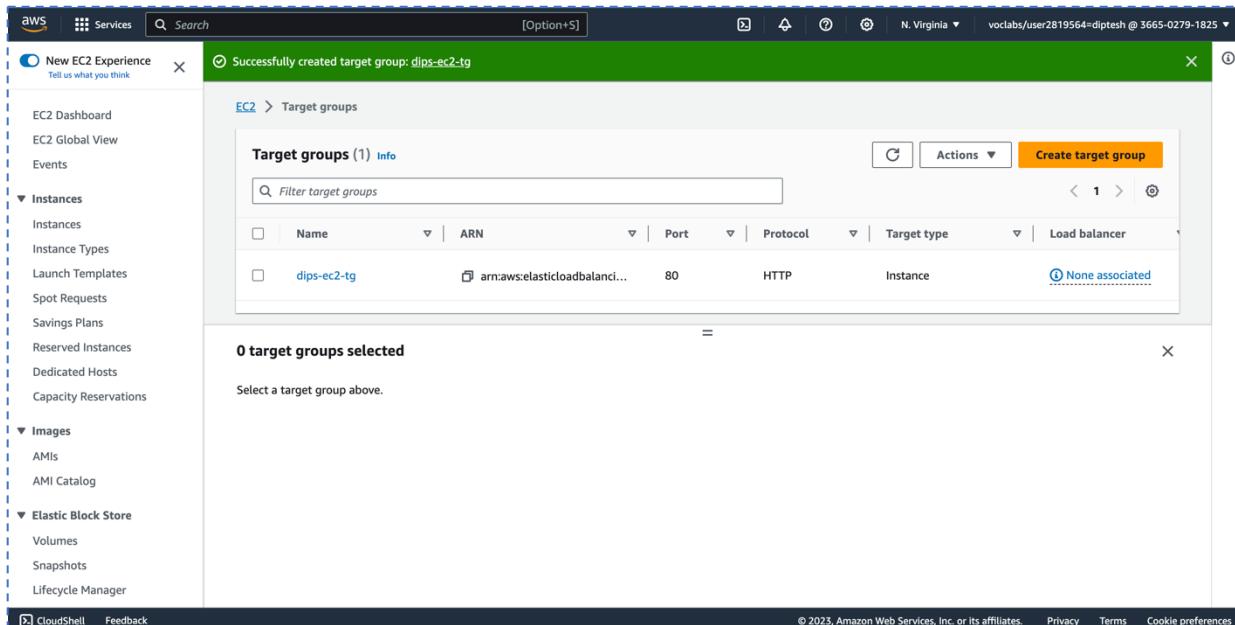
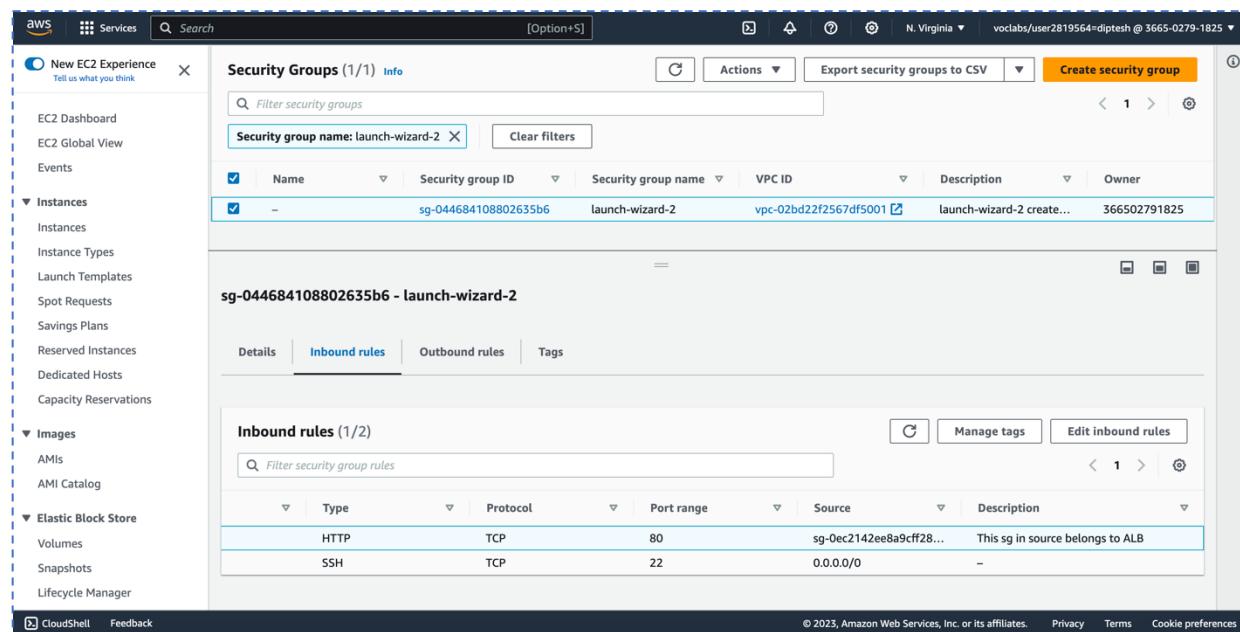
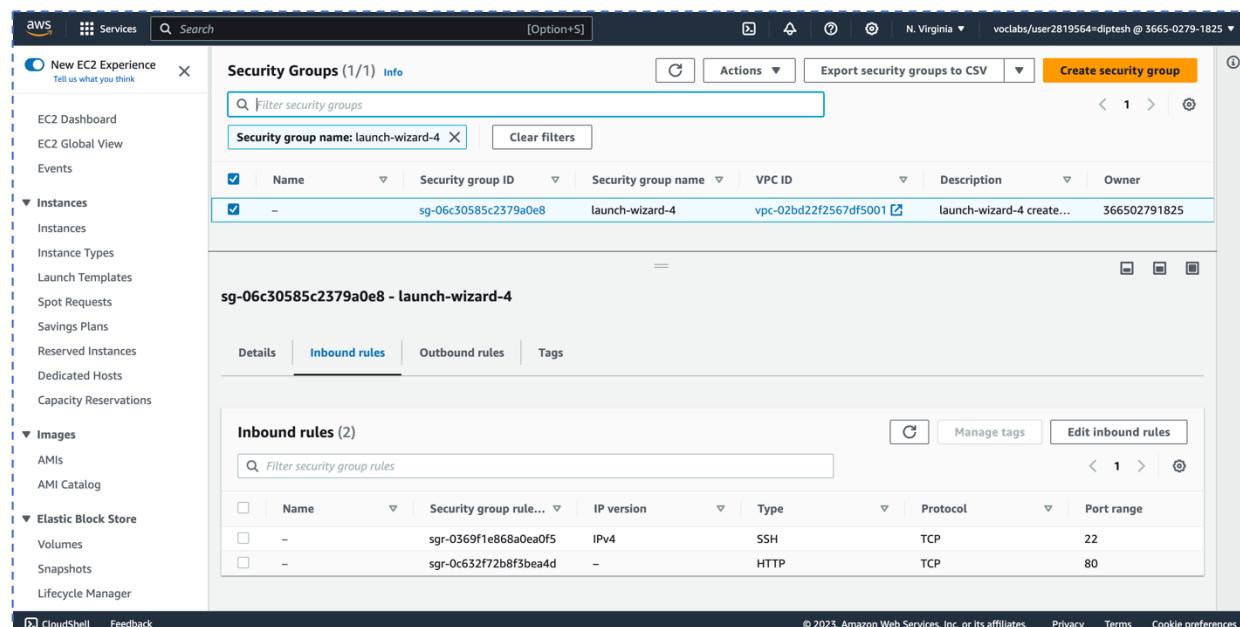


Image -10: Created target groups.



The screenshot shows the AWS EC2 Security Groups page. The security group 'sg-044684108802635b6 - launch-wizard-2' is selected. The Inbound rules section displays two entries:

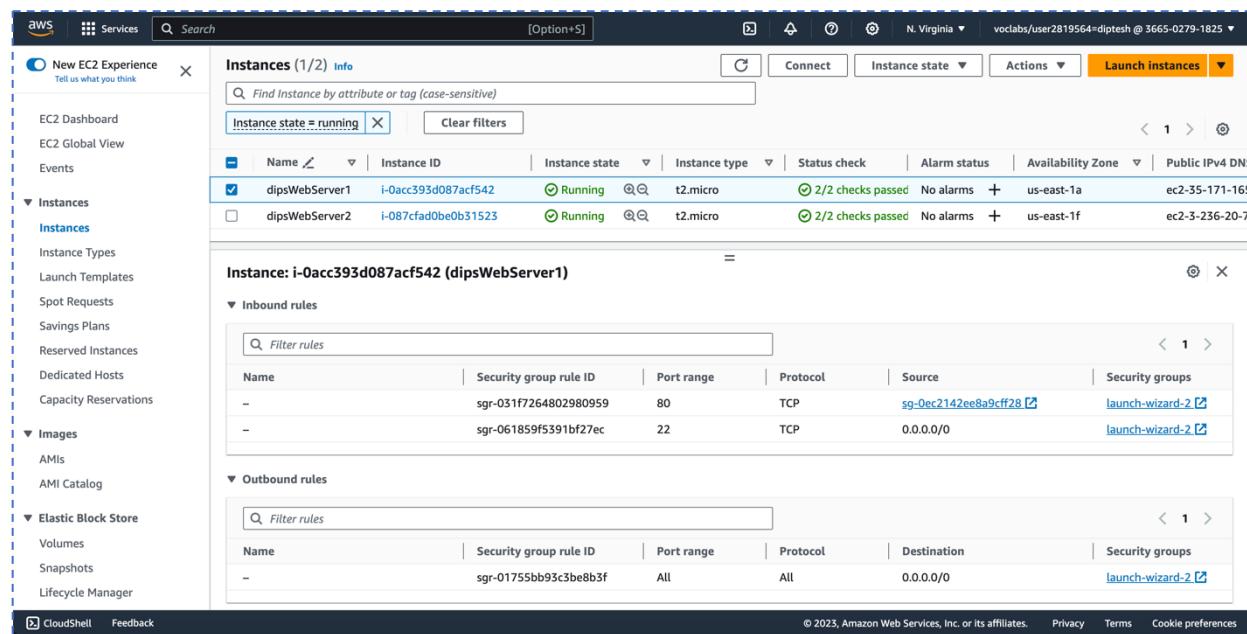
Type	Protocol	Port range	Source	Description
HTTP	TCP	80	sg-0ec2142ee8a9cff28...	This sg in source belongs to ALB
SSH	TCP	22	0.0.0.0/0	-

Image -11: Updated Security group for first EC2.


The screenshot shows the AWS EC2 Security Groups page. The security group 'sg-06c30585c2379a0e8 - launch-wizard-4' is selected. The Inbound rules section displays three entries:

Name	IP version	Type	Protocol	Port range
sgr-0369f1e868a0ea0f5	IPv4	SSH	TCP	22
sgr-0c632f72b8f3bea4d	-	HTTP	TCP	80

Image -12: Updated Security group for second EC2.



**Instances (1/2) Info**

Find Instance by attribute or tag (case-sensitive)

Instance state = running

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input checked="" type="checkbox"/> dipsWebServer1	i-0acc393d087acf542	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1a	ec2-35-171-165
<input type="checkbox"/> dipsWebServer2	i-087cfad0be0b31523	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1f	ec2-3-236-20-7

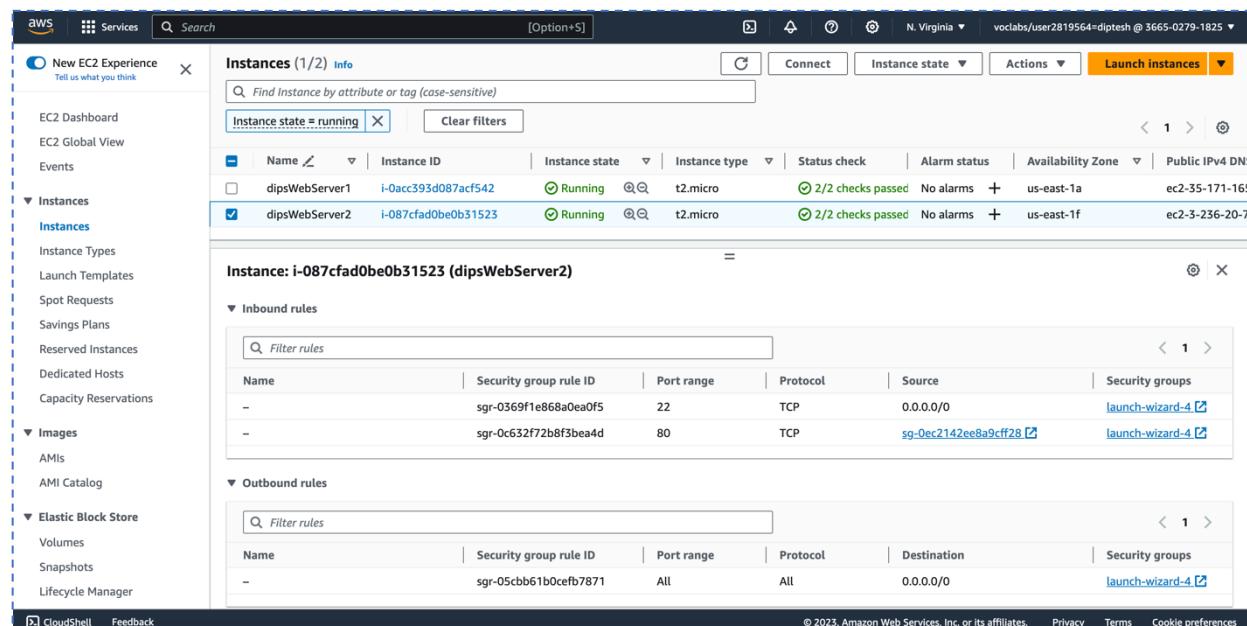
**Instance: i-0acc393d087acf542 (dipsWebServer1)**

Inbound rules

Name	Security group rule ID	Port range	Protocol	Source	Security groups
-	sgr-031f7264802980959	80	TCP	sg-0ec2142ee8a9cff28	launch-wizard-2
-	sgr-061859f5391bf27ec	22	TCP	0.0.0.0/0	launch-wizard-2

Outbound rules

Name	Security group rule ID	Port range	Protocol	Destination	Security groups
-	sgr-0175bb93c3be8b3f	All	All	0.0.0.0/0	launch-wizard-2

Image -13: Updated the inbound rules of first EC2 instance.


**Instances (1/2) Info**

Find Instance by attribute or tag (case-sensitive)

Instance state = running

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input type="checkbox"/> dipsWebServer1	i-0acc393d087acf542	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1a	ec2-35-171-165
<input checked="" type="checkbox"/> dipsWebServer2	i-087cfad0be0b31523	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1f	ec2-3-236-20-7

**Instance: i-087cfad0be0b31523 (dipsWebServer2)**

Inbound rules

Name	Security group rule ID	Port range	Protocol	Source	Security groups
-	sgr-0369f1e868a0ea0f5	22	TCP	0.0.0.0/0	launch-wizard-4
-	sgr-0c632f72b8f3bea4d	80	TCP	sg-0ec2142ee8a9cff28	launch-wizard-4

Outbound rules

Name	Security group rule ID	Port range	Protocol	Destination	Security groups
-	sgr-05ccb61b0cefb7871	All	All	0.0.0.0/0	launch-wizard-4

Image -14: Updated the inbound rules of second EC2 instance.

- Task 2: Practice Listener Rules with Lambdas

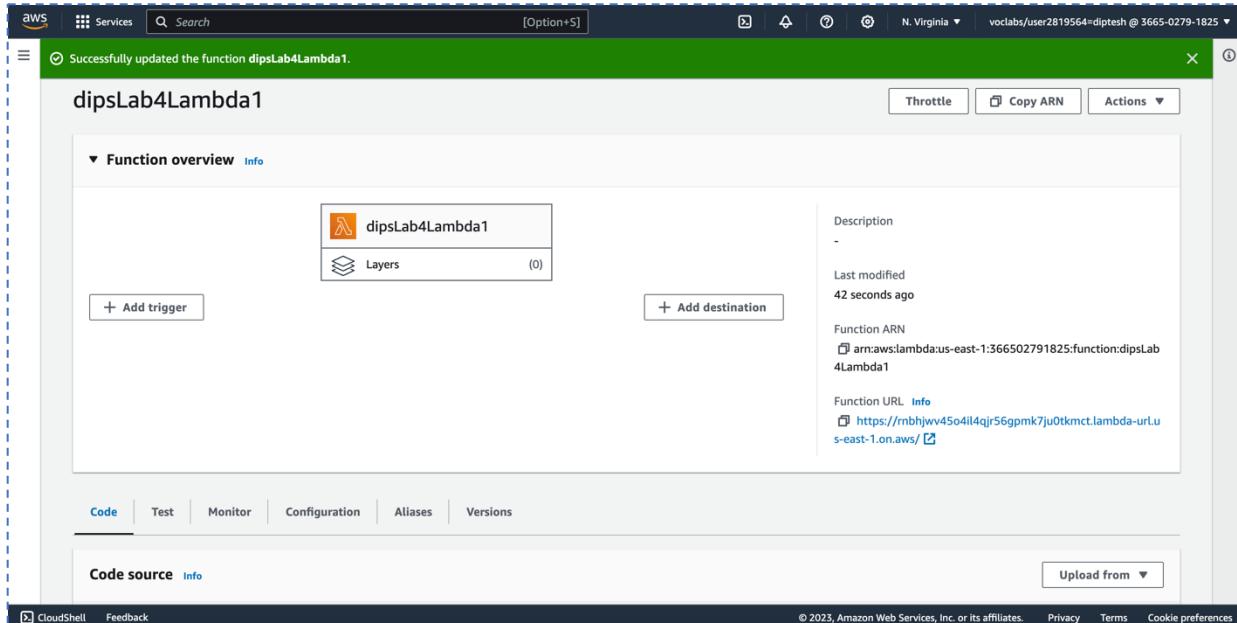


Image -15: Created & configured the first lambda function URL.

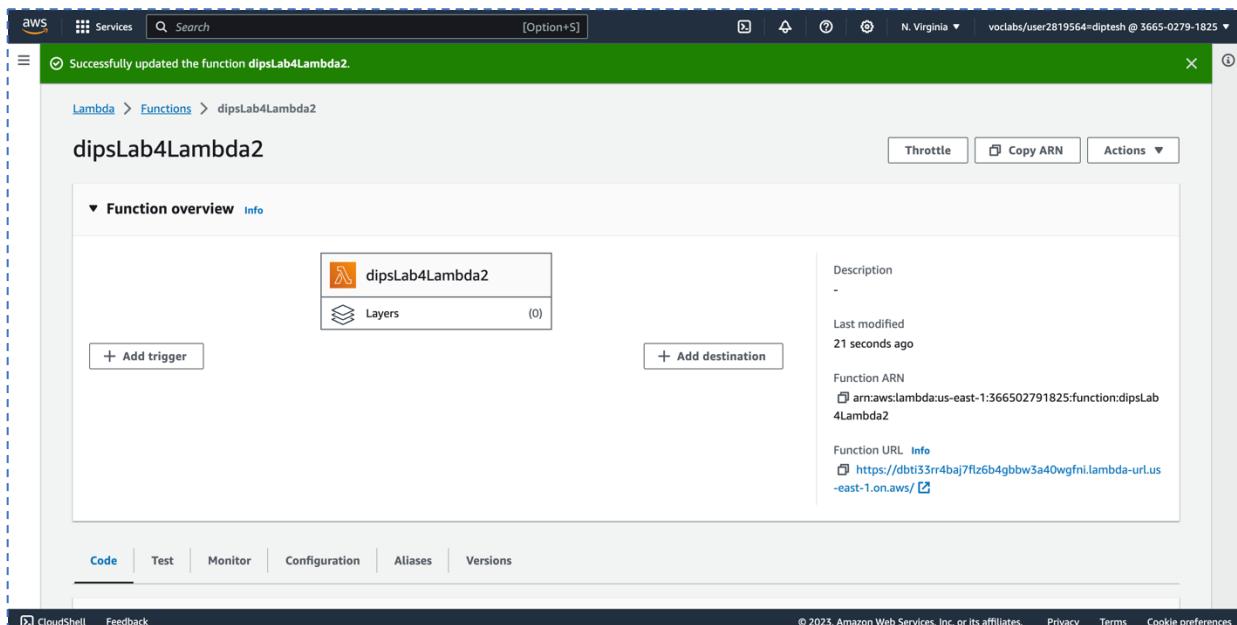
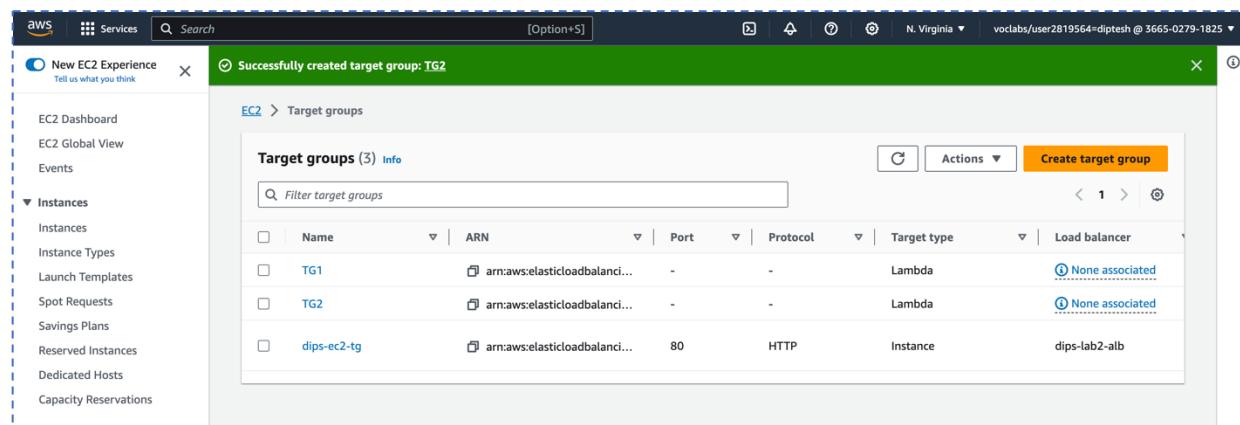


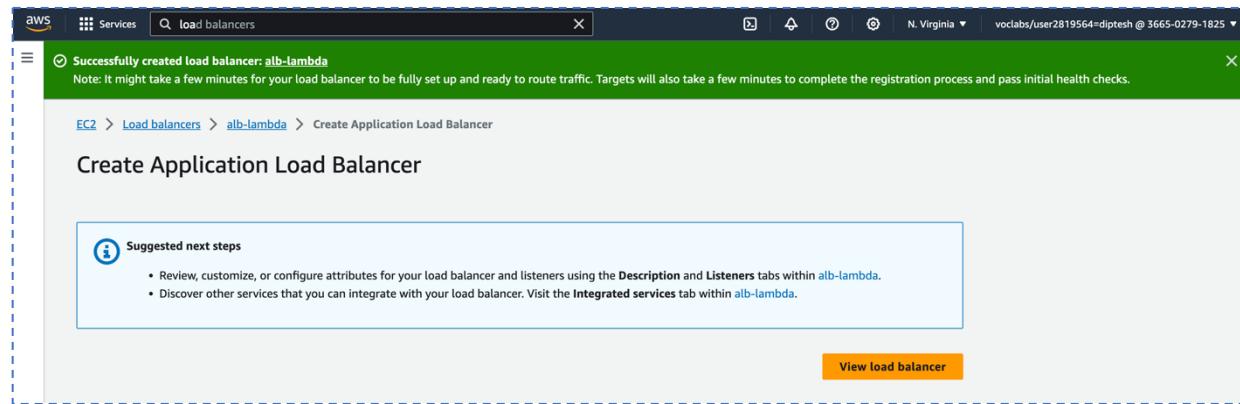
Image -16: Created & configured the second lambda function URL.



The screenshot shows the AWS EC2 Target Groups page. A success message at the top says "Successfully created target group: TG2". The main table lists three target groups:

Name	ARN	Port	Protocol	Target type	Load balancer
TG1	arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/TG1/7890123456789012	-	-	Lambda	(None)
TG2	arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/TG2/7890123456789012	-	-	Lambda	(None)
dips-ec2-tg	arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/dips-ec2-tg/7890123456789012	80	HTTP	Instance	dips-lab2-alb

Image - 17: Two target groups created for Lambda function.

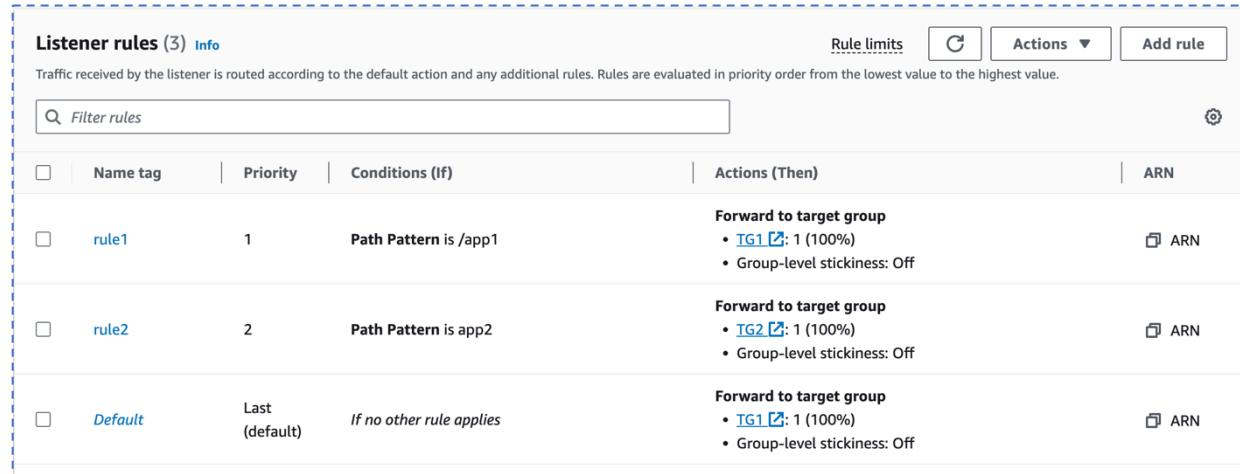


The screenshot shows the AWS Load Balancers page. A success message at the top says "Successfully created load balancer: alb-lambda". The main section shows the "Create Application Load Balancer" configuration. A "Suggested next steps" box contains:

- Review, customize, or configure attributes for your load balancer and listeners using the Description and Listeners tabs within alb-lambda.
- Discover other services that you can integrate with your load balancer. Visit the Integrated services tab within alb-lambda.

A "View load balancer" button is at the bottom right.

Image - 18: Created a new load balancer.



The screenshot shows the AWS Listener Rules page. A success message at the top says "Successfully created listener rule: rule1". The main table lists three rules:

Name tag	Priority	Conditions (If)	Actions (Then)	ARN
rule1	1	Path Pattern is /app1	<b>Forward to target group</b> • TG1: 1 (100%) • Group-level stickiness: Off	arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/12345678901234567890123456789012
rule2	2	Path Pattern is app2	<b>Forward to target group</b> • TG2: 1 (100%) • Group-level stickiness: Off	arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/12345678901234567890123456789012
Default	Last (default)	If no other rule applies	<b>Forward to target group</b> • TG1: 1 (100%) • Group-level stickiness: Off	arn:aws:elasticloadbalancing:us-east-1:123456789012:listener-rule/12345678901234567890123456789012

Image - 19: Created new listener rules with proper condition.

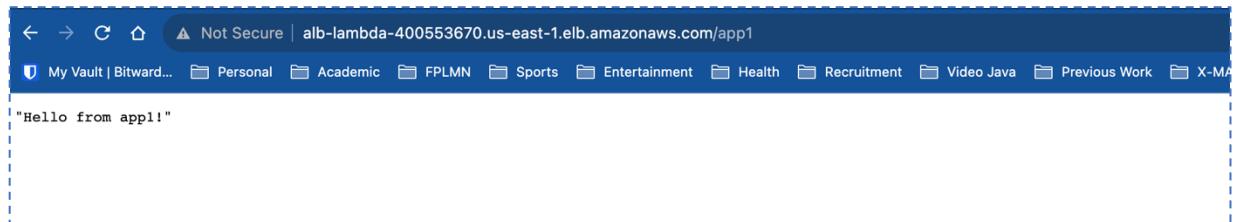


Image - 20: Browsing lambda function 1 via ALB slash app1.

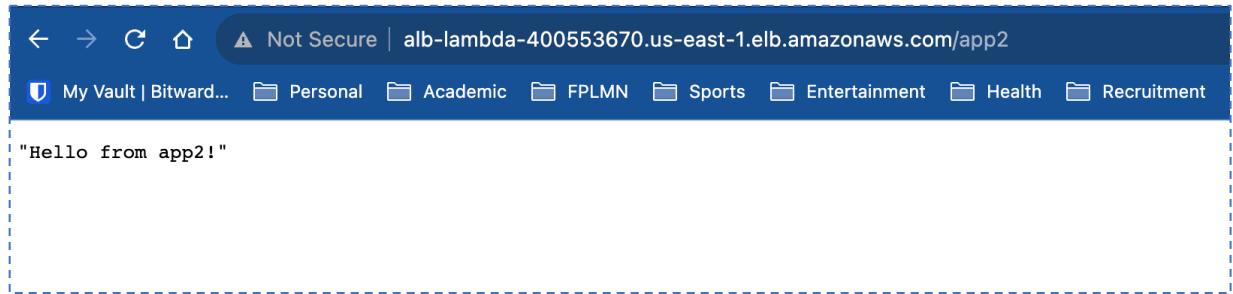


Image - 21: Browsing lambda function 2 via ALB slash app2.

- **Task 3: Run web servers behind NLB**

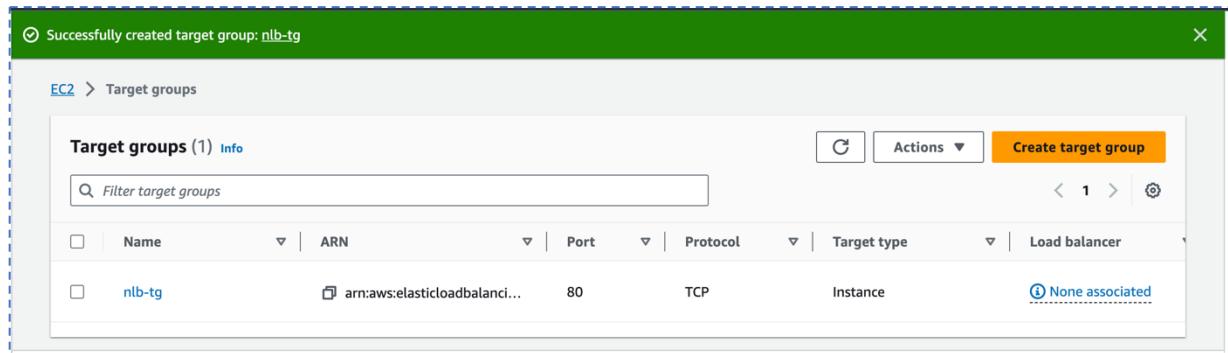


Image - 22: Created a new target group for NLB.

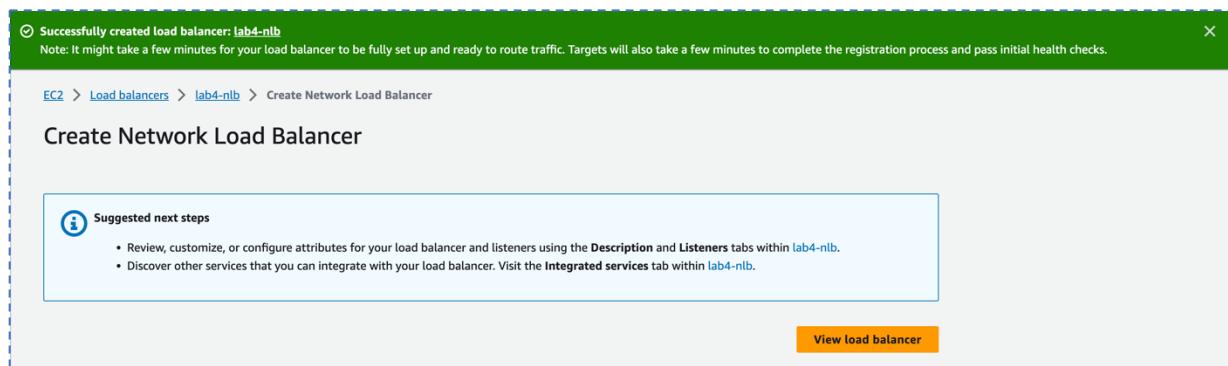
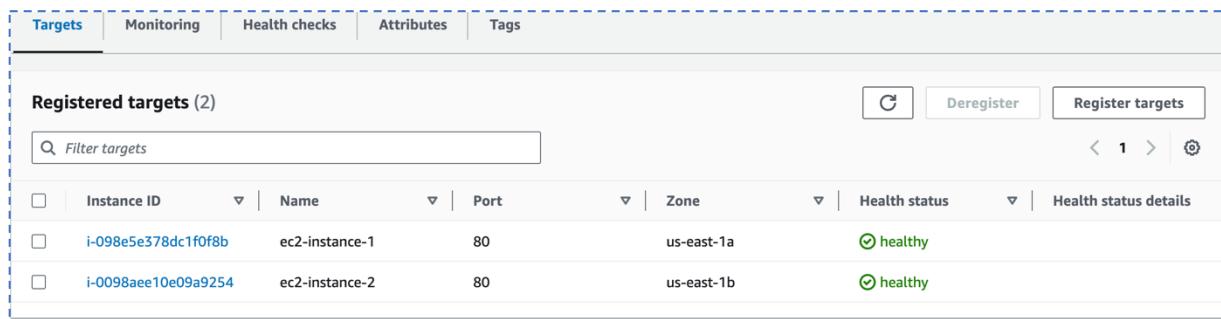


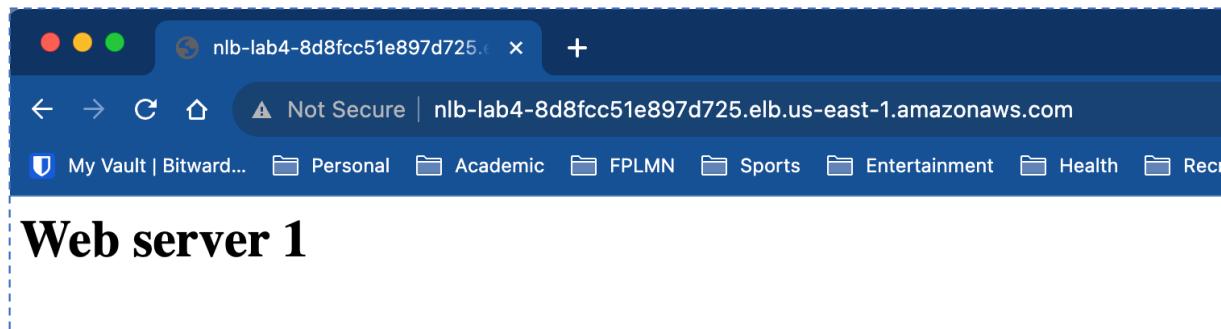
Image - 23: Created a new load balancer.



The screenshot shows the AWS Lambda Targets console. At the top, there are tabs for Targets, Monitoring, Health checks, Attributes, and Tags. Below the tabs, it says "Registered targets (2)". There is a search bar labeled "Filter targets". On the right side of the table are buttons for "Deregister" and "Register targets". There are also navigation arrows and a refresh icon.

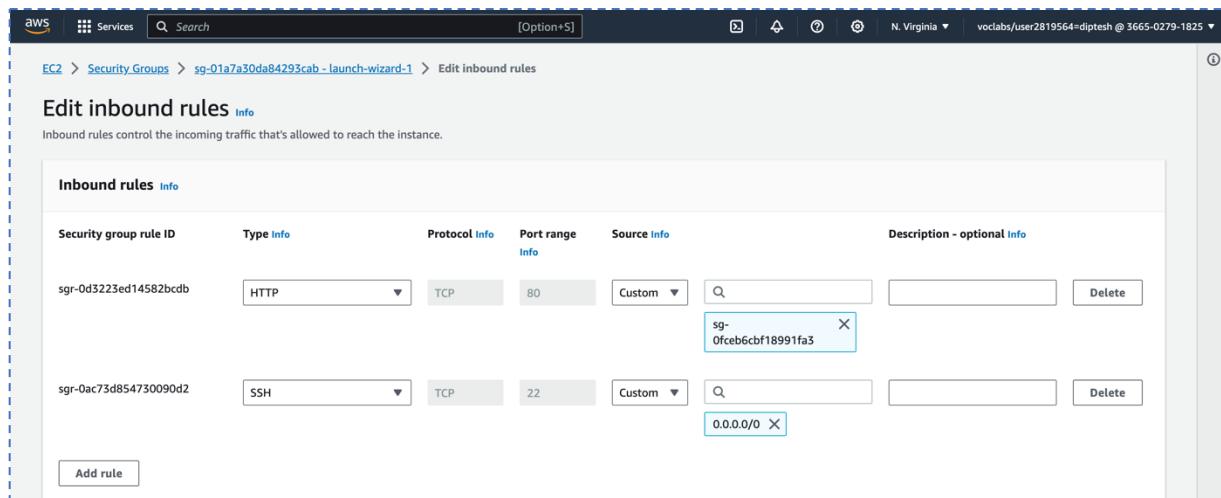
Instance ID	Name	Port	Zone	Health status	Health status details
i-098e5e378dc1f0f8b	ec2-instance-1	80	us-east-1a	<span>healthy</span>	
i-0098aee10e09a9254	ec2-instance-2	80	us-east-1b	<span>healthy</span>	

Image - 23: EC2 instances registered in the NLB target group.



The screenshot shows a web browser window with the URL "nlb-lab4-8d8fcc51e897d725.elb.us-east-1.amazonaws.com". The page content is "Web server 1".

Image - 23: Browsing application via NLB.



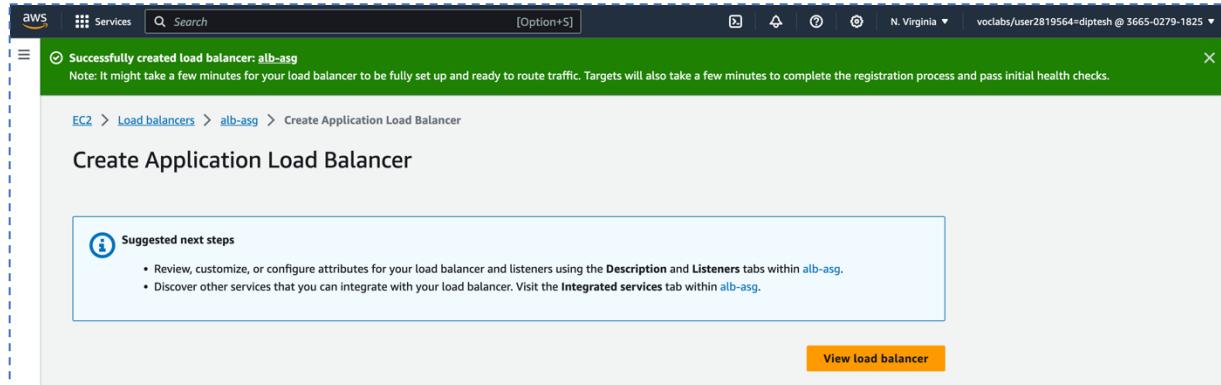
The screenshot shows the AWS Security Groups console. It is navigating through EC2 > Security Groups > sg-01a7a30da84293cab - launch-wizard-1 > Edit inbound rules. The title is "Edit inbound rules". It says "Inbound rules control the incoming traffic that's allowed to reach the instance." Below this, there is a table for "Inbound rules".

Security group rule ID	Type	Protocol	Port range	Source	Description
sgr-0d3223ed14582bcd8	HTTP	TCP	80	Custom	sg-0fce6cbf18991fa3
sgr-0ac73d854730090d2	SSH	TCP	22	Custom	0.0.0.0/0

At the bottom left, there is a button "Add rule".

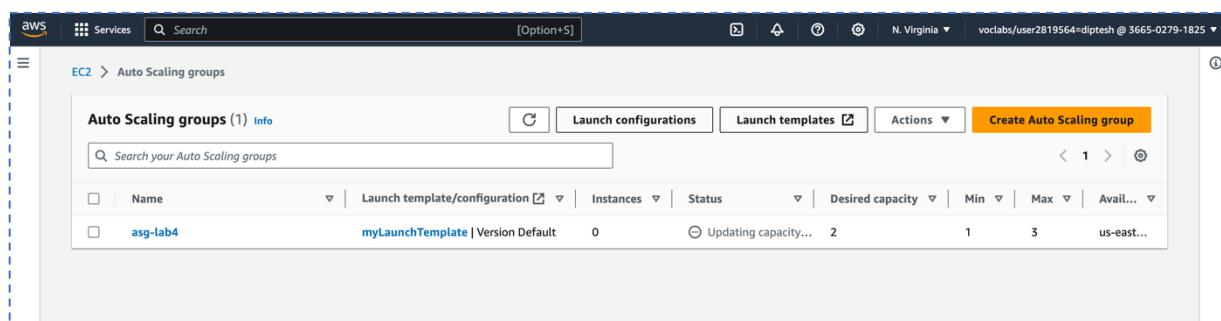
Image - 23: Updated the inbound rules of the EC2 security group to allow access only from NLB and no direct access.

- Task 4: Run the web servers behind the ALB in ASG



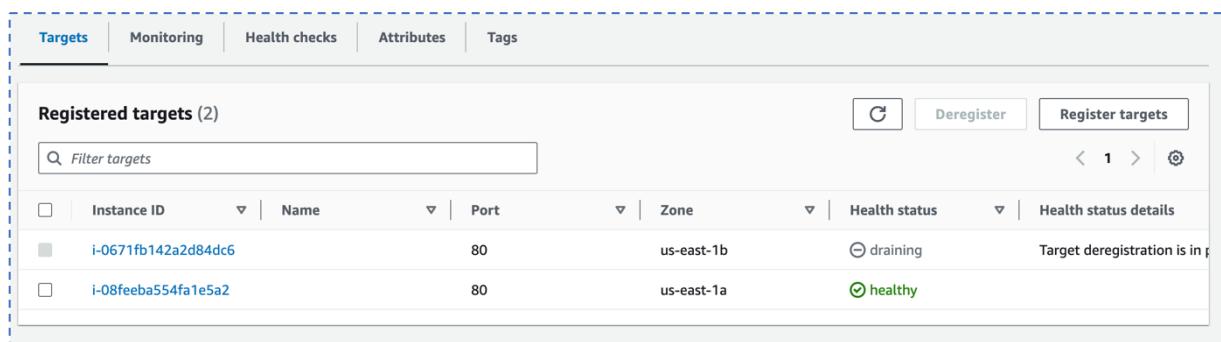
The screenshot shows the AWS EC2 Load Balancers console. A green success message at the top states: "Successfully created load balancer: alb-asg. Note: It might take a few minutes for your load balancer to be fully set up and ready to route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks." Below this, the "Create Application Load Balancer" page is displayed, featuring a "Suggested next steps" box with instructions to review attributes and discover integrations, and a prominent orange "View load balancer" button.

Image - 24: Created a new load balancer.



The screenshot shows the AWS Auto Scaling groups console. It displays a table of one Auto Scaling group: "asg-lab4". The group uses the launch template "myLaunchTemplate" and has a default version. The status shows "Updating capacity..." with values 2, 1, 3, and "us-east...". An orange "Create Auto Scaling group" button is visible at the top right.

Image - 25: Created a new auto scaling group.



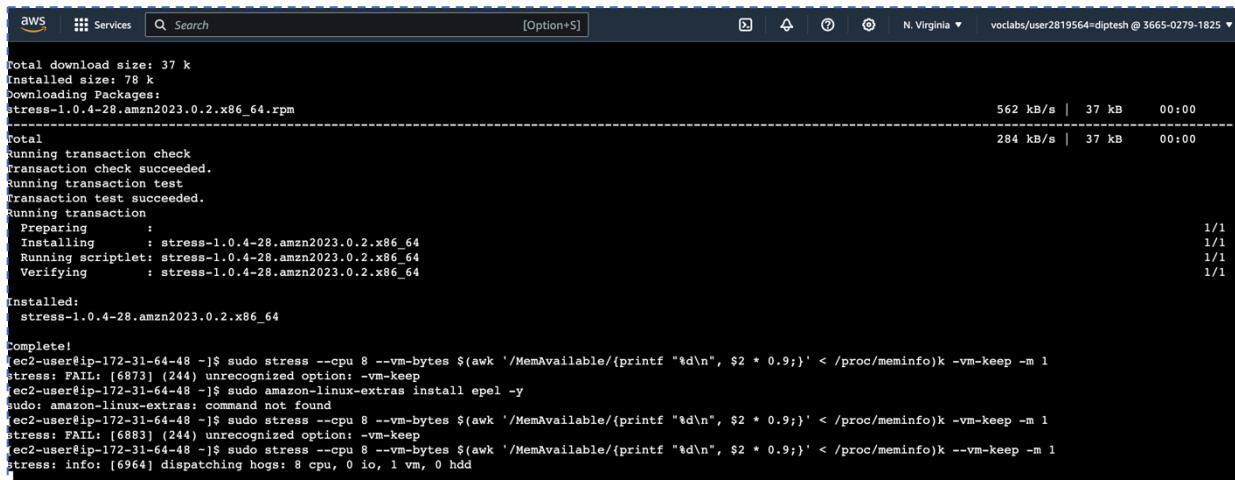
The screenshot shows the AWS Targets console. It lists two registered targets: an Amazon Linux instance with Instance ID i-0671fb142a2d84dc6 and Port 80, located in Zone us-east-1b, with a Health status of "draining"; and another instance with Instance ID i-08feeba554fa1e5a2 and Port 80, located in Zone us-east-1a, with a Health status of "healthy". A "Register targets" button is visible at the top right.

Image - 26: Created a new load balancer.



The screenshot shows a web browser window with the URL "alb-asg-486099335.us-east-1.elb.amazonaws.com". The page content reads "Hello YourName from ip-172-31-32-226.ec2.internal".

Image - 27: Created a new load balancer.



```

aws Services Search [Option+S] N. Virginia vclabs/user2819564=diptesh@3665-0279-1825

Total download size: 37 k
Installed size: 78 k
Downloading Packages:
stress-1.0.4-28.amzn2023.0.2.x86_64.rpm
562 kB/s | 37 kB 00:00
Total
running transaction check
Transaction check succeeded.
running transaction test
Transaction test succeeded.
running transaction
Preparing : 1/1
Installing : stress-1.0.4-28.amzn2023.0.2.x86_64 1/1
Running scriptlets: stress-1.0.4-28.amzn2023.0.2.x86_64 1/1
Verifying : stress-1.0.4-28.amzn2023.0.2.x86_64 1/1

Installed:
stress-1.0.4-28.amzn2023.0.2.x86_64

Complete!
[ec2-user@ip-172-31-64-48 ~]$ sudo stress --cpu 8 --vm-bytes $(awk '/MemAvailable/{printf "%d\n", $2 * 0.9;}' < /proc/meminfo)k -vm-keep -m 1
stress: FAIL: [6873] (244) unrecognized option: -vm-keep
[ec2-user@ip-172-31-64-48 ~]$ sudo amazon-linux-extras install epel -y
sudo: amazon-linux-extras: command not found
[ec2-user@ip-172-31-64-48 ~]$ sudo stress --cpu 8 --vm-bytes $(awk '/MemAvailable/{printf "%d\n", $2 * 0.9;}' < /proc/meminfo)k -vm-keep -m 1
stress: FAIL: [6883] (244) unrecognized option: -vm-keep
[ec2-user@ip-172-31-64-48 ~]$ sudo stress --cpu 8 --vm-bytes $(awk '/MemAvailable/{printf "%d\n", $2 * 0.9;}' < /proc/meminfo)k --vm-keep -m 1
stress: info: [6964] dispatching hogs: 8 cpu, 0 io, 1 vm, 0 hdd

```

Image -28: Simulating CPU and memory utilization for checking auto scaling.