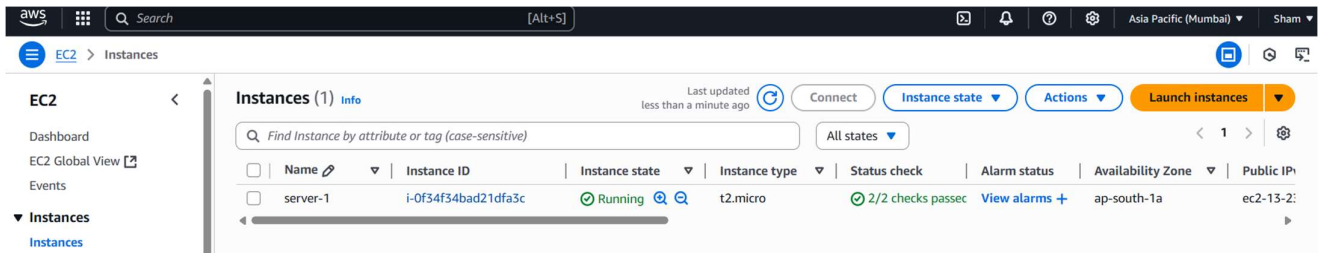


Task: Run a docker container with port mapping and access through load balancer.

Creating two separate EC2 Instances and installing docker.

1. Server-1, OS: Amazon Linux

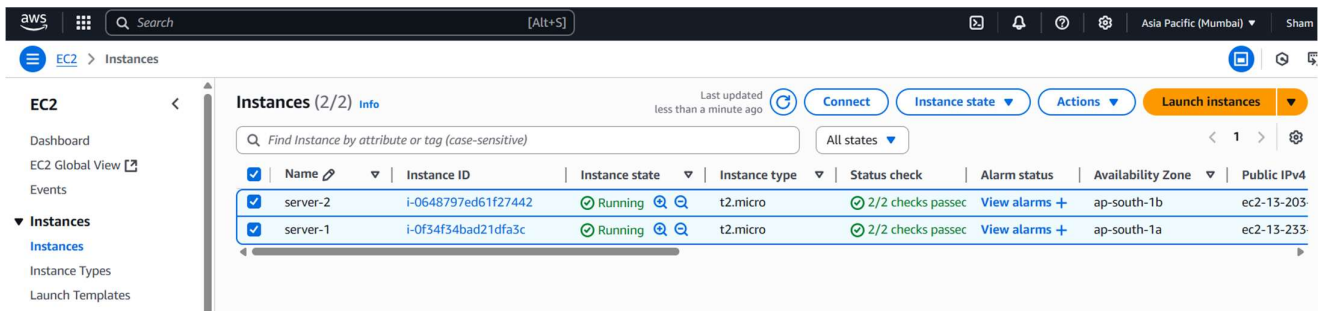


The screenshot shows the AWS Management Console 'Instances' page. On the left, the 'EC2' menu is expanded, showing 'Instances' as the selected option. The main panel displays 'Instances (1)' with a search bar and a table of instances. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IP. One instance, 'server-1', is listed with ID 'i-0f34f34bad21dfa3c', state 'Running', type 't2.micro', and status '2/2 checks passed'. Buttons for 'Connect', 'Instance state', 'Actions', and 'Launch instances' are visible at the top right.

```
[root@ip-172-31-42-62 ~]# docker --version
Docker version 25.0.8, build 0bab007
[root@ip-172-31-42-62 ~]# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled;
   Active: active (running) since Tue 2025-05-27 08:52:03 UTC; 27s ago
   TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
    Process: 27877 ExecStartPre=/bin/mkdir -p /run/docker (code=exited
    Process: 27878 ExecStartPre=/usr/libexec/docker/docker-setup-run-ti
   Main PID: 27879 (dockerd)
     Tasks: 7
    Memory: 37.9M
       CPU: 267ms
    CGroup: /system.slice/docker.service
            └─27879 /usr/bin/dockerd -H fd:// --containerd=/run/conta

May 27 08:52:02 ip-172-31-42-62.ap-south-1.compute.internal systemd[1]
May 27 08:52:02 ip-172-31-42-62.ap-south-1.compute.internal dockerd[27
May 27 08:52:02 ip-172-31-42-62.ap-south-1.compute.internal dockerd[27
```

2. Server-2, OS: Ubuntu



The screenshot shows the AWS Management Console 'Instances' page with two instances. The table now lists 'server-1' and 'server-2'. 'server-2' has ID 'i-0648797ed61f27442' and is also in a 'Running' state with '2/2 checks passed'. The 'Launch instances' button is highlighted in orange.

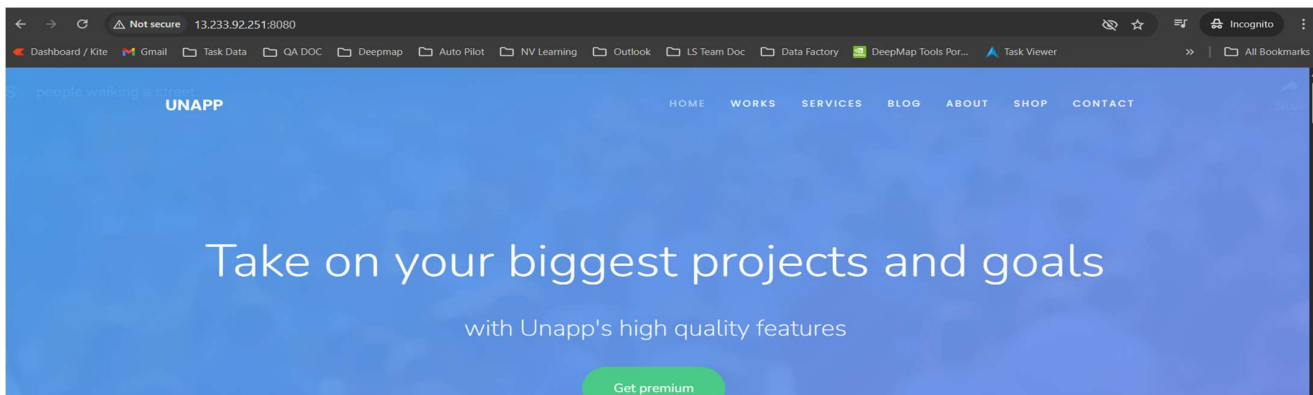
```
root@ip-172-31-9-165: ~
root@ip-172-31-9-165:~# docker --version
Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1
root@ip-172-31-9-165:~# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled;
   Active: active (running) since Tue 2025-05-27 08:53:43 UTC;
   TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
    Main PID: 1961 (dockerd)
     Tasks: 8
    Memory: 60.5M (peak: 60.6M)
       CPU: 303ms
    CGroup: /system.slice/docker.service
            └─1961 /usr/bin/dockerd -H fd:// --containerd=/run/c

May 27 08:53:43 ip-172-31-9-165 systemd[1]: Starting docker.servi
May 27 08:53:43 ip-172-31-9-165 dockerd[1961]: time="2025-05-27T0
```

Running container on each EC2 server with port mapping.

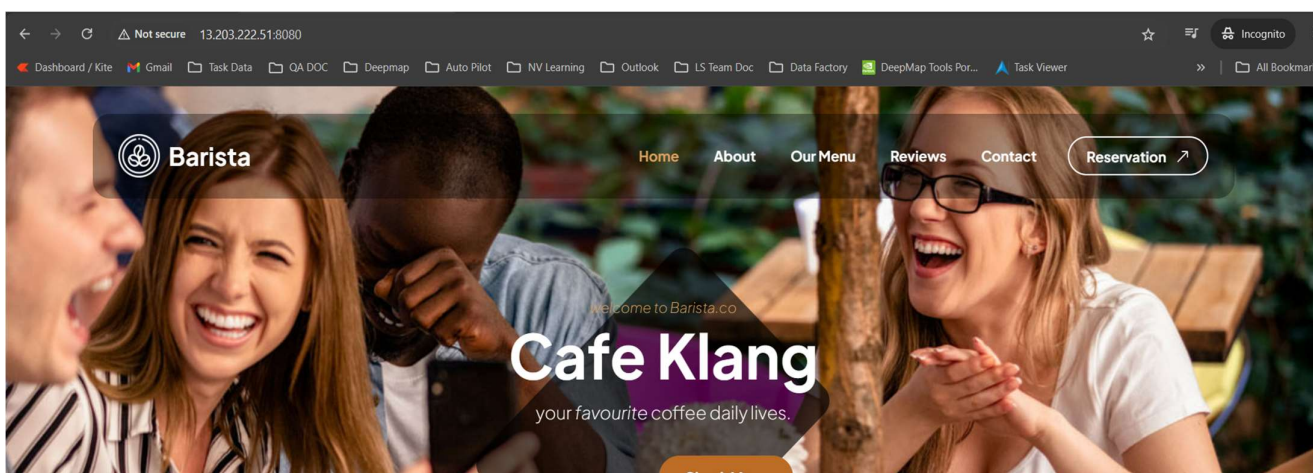
1. Server-1

```
root@ip-172-31-42-62:~  
[root@ip-172-31-42-62 ~]# docker images  
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE  
sham9394/drupal-webapp v2           94775ac25fe7     5 months ago    552MB  
[root@ip-172-31-42-62 ~]# docker ps  
CONTAINER ID        IMAGE               COMMAND            CREATED          STATUS          PORTS  
c6fab9acb7a1       sham9394/drupal-webapp:v2 "docker-php-entryp... 10 minutes ago   Up 10 minutes   0.0.0.0:8080->80/tcp, :::8080->80/tcp  
wonderful_chatterjee  
[root@ip-172-31-42-62 ~]# |
```



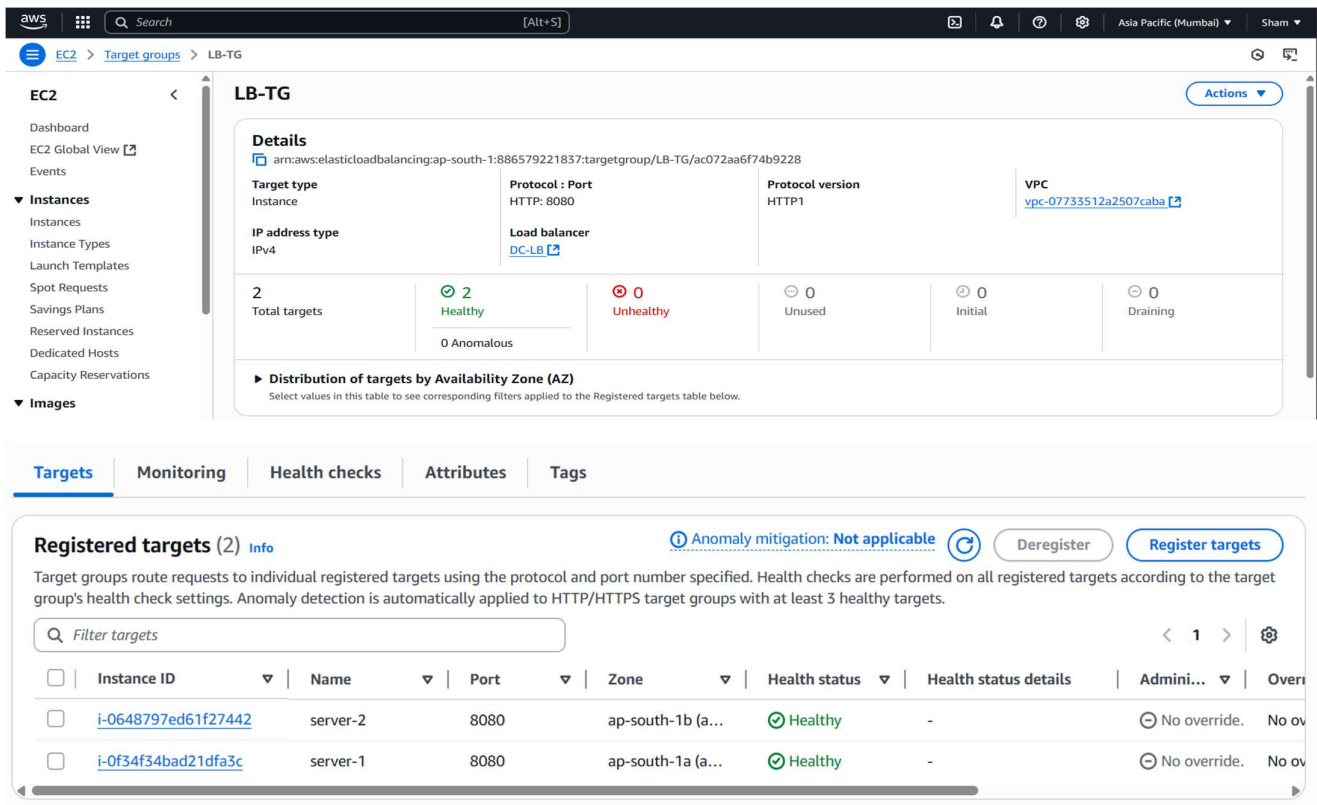
2. Server-2

```
root@ip-172-31-9-165: ~  
root@ip-172-31-9-165:~# docker images  
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE  
sham9394/web-app    01           1e70083e3add     10 months ago    193MB  
root@ip-172-31-9-165:~# docker ps  
CONTAINER ID        IMAGE               COMMAND            CREATED          STATUS          PORTS  
9824e49118e9       sham9394/web-app:01 "/docker-entrypoin... 6 minutes ago    Up 6 minutes    0.0.0.0:8080->80/tcp, :::8080->80/tcp  
magical_ganguly  
root@ip-172-31-9-165:~# |
```



Access application through Load balancer

1. Creating Target Group.



The screenshot shows the AWS Management Console interface for a Target Group named 'LB-TG'. The left sidebar contains navigation links for EC2, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, and Images. The main content area displays the 'Details' tab for the Target Group. The details include the ARN, Target type (Instance), Protocol (HTTP), Port (8080), Protocol version (HTTP1), VPC (vpc-07733512a2507caba), IP address type (IPv4), and Load balancer (DC-LB). A summary row shows 2 Total targets, 2 Healthy, 0 Unhealthy, 0 Unused, 0 Initial, and 0 Draining. Below this is a section for 'Distribution of targets by Availability Zone (AZ)' with a note to select values in the table below.

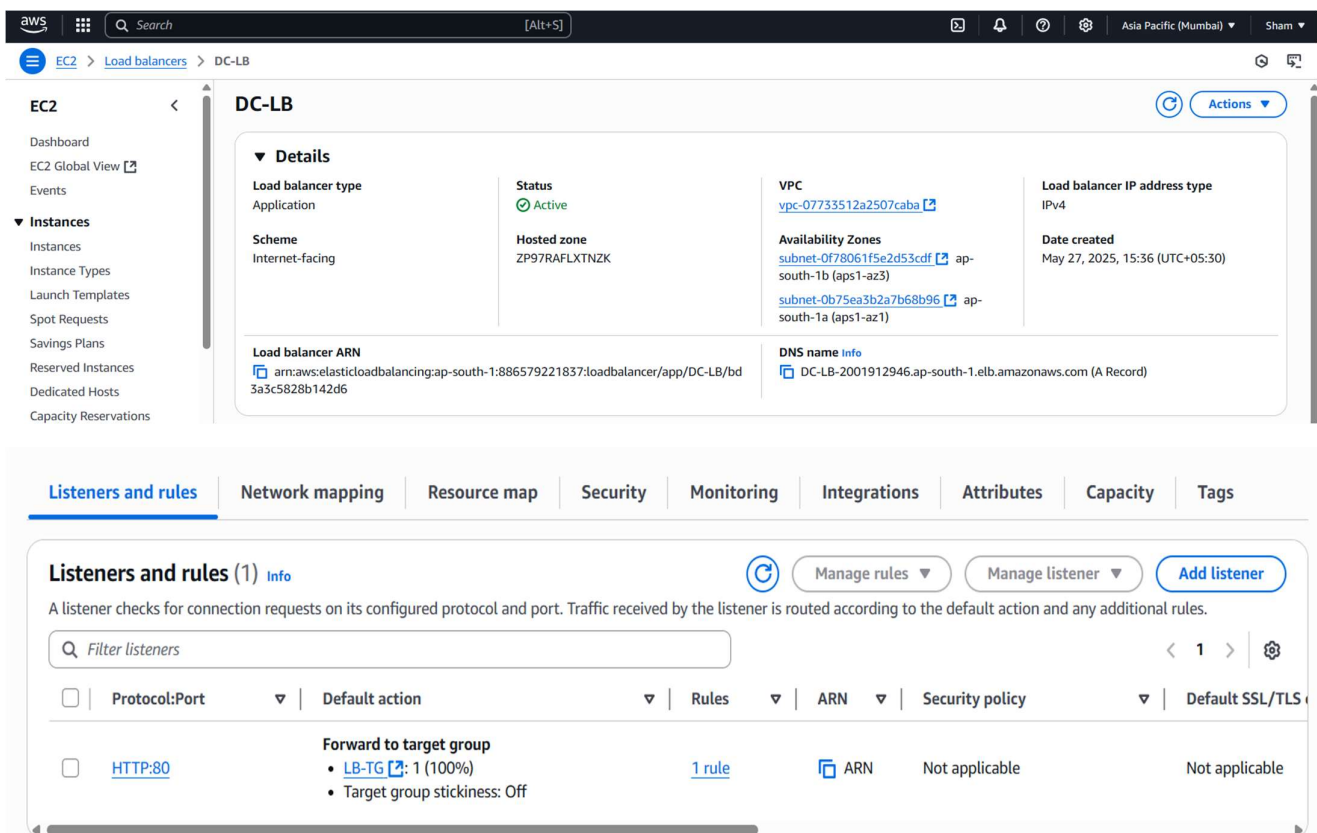
Registered targets (2) Info [Anomaly mitigation: Not applicable](#) [Deregister](#) [Register targets](#)

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

Filter targets

<input type="checkbox"/>	Instance ID	Name	Port	Zone	Health status	Health status details	Admini...	Overv
<input type="checkbox"/>	i-0648797ed61f27442	server-2	8080	ap-south-1b (a...	Healthy	-	No override.	No ov
<input type="checkbox"/>	i-0f34f34bad21dfa3c	server-1	8080	ap-south-1a (a...	Healthy	-	No override.	No ov

2. Creating Load Balancer.



The screenshot shows the AWS Management Console interface for a Load Balancer named 'DC-LB'. The left sidebar contains navigation links for EC2, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, and Images. The main content area displays the 'Details' tab for the Load Balancer. The details include the Load balancer type (Application), Status (Active), VPC (vpc-07733512a2507caba), Load balancer IP address type (IPv4), Scheme (Internet-facing), Hosted zone (ZP97RAFLXTNZK), Availability Zones (subnet-0f78061f5e2d53cdf, subnet-0b75ea3b2a7b68b96), Date created (May 27, 2025, 15:36 (UTC+05:30)), Load balancer ARN (arn:aws:elasticloadbalancing:ap-south-1:886579221837:loadbalancer/app/DC-LB/bd3a3c5828b142d6), and DNS name info (DC-LB-2001912946.ap-south-1.elb.amazonaws.com (A Record)).

Listeners and rules (1) Info [Manage rules](#) [Manage listener](#) [Add listener](#)

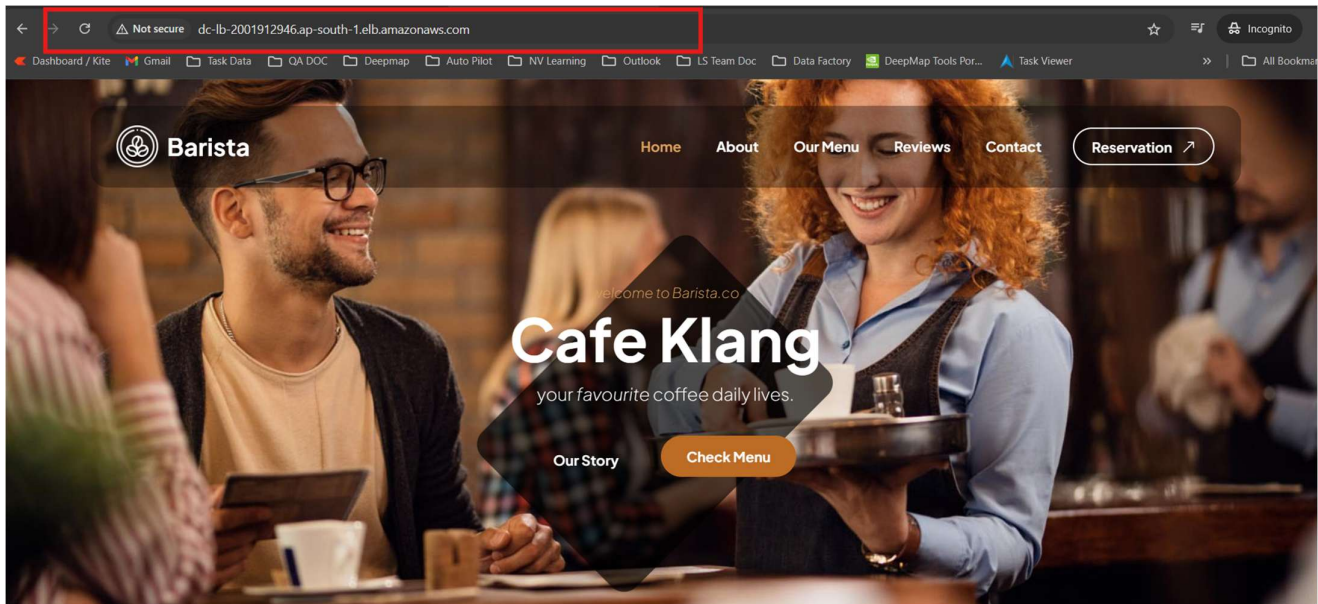
A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Filter listeners

<input type="checkbox"/>	Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS
<input type="checkbox"/>	HTTP:80	Forward to target group <ul style="list-style-type: none">LB-TG: 1 (100%)Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable

Accessing web application via Load Balancer:

1. Initial Output from Server-2



2. After refresh output from Server-1.

