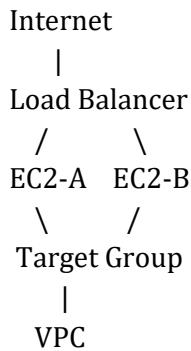


# Highly Available Web Application Deployment Using AWS

## Objective:

To deploy a simple web application on two EC2 instances located in different Availability Zones, fronted by an Application Load Balancer (ALB), ensuring high availability — if one EC2 instance fails, the other serves traffic seamlessly.

## Architecture Diagram:



## Step-by-Step Implementation:

### Step 1: Create a VPC

CIDR Block: 10.0.0.0/16

The screenshot shows the AWS VPC dashboard with the following details:

- VPC dashboard**: The main navigation bar.
- Virtual private cloud**: A sidebar menu with options like Subnets, Route tables, Internet gateways, Egress-only internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways, and Peering connections.
- Your VPCs (1/2)**: The main content area. It shows a table with two rows:

Name	VPC ID	State	Block Public Access	IPv4 CIDR
vpc-for-web-application-deployment	vpc-01756043392ef4719	Available	Off	10.0.0.0/16
	vpc-08e490d66dd663677	Available	Off	172.31.0.0/16
- vpc-01756043392ef4719 / vpc-for-web-application-deployment**: A detailed view of the selected VPC. It shows the following configuration:

Details	State	Block Public Access	DNS hostnames
VPC ID: vpc-01756043392ef4719	Available	Off	Disabled
DNS resolution: Enabled	Tenancy: default	DHCP option set: dopt-01ae084900474afe9	Main route table: rtb-092e4b11c7d301aae
Main network ACL: acl-028884102633fad14	Default VPC: No	IPv4 CIDR: 10.0.0.0/16	IPv6 pool: -

## Step 2: Create Two Public Subnets

Subnet-1 | us-east-1a | 10.0.1.0/24

Subnet-2 | us-east-1b | 10.0.2.0/24

The screenshot shows the AWS VPC Subnets page. On the left, there's a navigation sidebar with options like EC2 Global View, Filter by VPC, Virtual private cloud (Your VPCs, Subnets), Security (Network ACLs, Security groups), and a main VPC dashboard. The main area displays a table of subnets:

Name	Subnet ID	State	VPC
web-deployment-1a	subnet-0299aaf60ff2d8e98	Available	vpc-01756043392ef4719   vpc-for-web-application-deployment
web-deployment-1b	subnet-0cdaceee13a9e660	Available	vpc-01756043392ef4719   vpc-for-web-application-deployment
-	subnet-013c9c9ef430f08da	Available	vpc-08e490d66dd663677
-	subnet-09a0a35dec79b6b2b	Available	vpc-08e490d66dd663677

Below the table, a specific subnet (subnet-0299aaf60ff2d8e98) is selected, showing its details:

Subnet ID	Subnet ARN	State	Block Public Access
subnet-0299aaf60ff2d8e98	arn:aws:ec2:ap-south-1:750311440127:subnet/subnet-0299aaf60ff2d8e98	Available	Off
IPv4 CIDR	IPv6 CIDR	IPv6 CIDR association ID	
10.0.1.0/24	-	-	
Availability Zone	Available IPv4 addresses	Network border group	VPC
ap-south-1a	249	ap-south-1	vpc-01756043392ef4719   vpc-for-web-application-deployment
Route table	Availability Zone ID	Default subnet	No

## Step 3: Create and Attach an Internet Gateway

Attach it to the VPC.

The screenshot shows the AWS Internet Gateways page. The left sidebar is identical to the previous VPC Subnets page. The main area displays a table of internet gateways:

Name	Internet gateway ID	State	VPC ID
igw-web-deployment	igw-051ebd3f56c46e6fa	Attached	vpc-01756043392ef4719   vpc-for-web-application-deployment
-	igw-0622073f5da35c7c	Attached	vpc-08e490d66dd663677

Below the table, a specific internet gateway (igw-051ebd3f56c46e6fa) is selected, showing its details:

igw-051ebd3f56c46e6fa / igw-web-deployment			
Details	Tags		
Details	Tags		
Internet gateway ID: igw-051ebd3f56c46e6fa	State: Attached	VPC ID: vpc-01756043392ef4719   vpc-for-web-application-deployment	Owner: 750311440127

## Step 4: Create a Route Table and Attach to Subnets

Add route: Destination: 0.0.0.0/0 → Target: Internet Gateway

Associate with both public subnets.

Route tables (1/3) [Info](#)

Name	Route table ID	Explicit subnet associations	Edge associations	Main
-	rtb-093bf42c4c203d8ab	-	-	Yes
-	rtb-092e4b11c7d301aae	-	-	Yes
<input checked="" type="checkbox"/> web-deployment-rt	rtb-08a0639fc9fdf48a4	2 subnets	-	No

**rtb-08a0639fc9fdf48a4 / web-deployment-rt**

- [Details](#)
- [Routes](#)
- [Subnet associations](#) **(2)** [Edit subnet associations](#)
- [Edge associations](#)
- [Route propagation](#)
- [Tags](#)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
web-deployment-1a	subnet-0299aaf60ff2d8e98	10.0.1.0/24	-
web-deployment-1b	subnet-0cdaceee13a9e660	10.0.2.0/24	-

## Step 5: Launch Two EC2 Instances (Web Servers) with Created VPC

AMI: Amazon Linux 2, Type: t2.micro, One in each subnet

Use this user data:

```
#!/bin/bash
sudo yum update -y
sudo yum install -y httpd
echo "Welcome to $(hostname) - Web Server" > /var/www/html/index.html
sudo systemctl start httpd
sudo systemctl enable httpd
```

Instances (1/2) [Info](#)

Name	Instance ID	Instance state	Instance type	Status check	Alarm
-	i-0ce4f75866ada822a	Running	t2.micro	2/2 checks passed	<a href="#">View</a>
<input checked="" type="checkbox"/> web-deployment-server-1	i-073790655b83f4e7c	Running	t2.micro	Initializing	<a href="#">View</a>

**i-073790655b83f4e7c (web-deployment-server-1)**

Instance ID <a href="#">i-073790655b83f4e7c</a>	Public IPv4 address <a href="#">65.2.34.136   open address</a>	Private IPv4 addresses 10.0.1.84
IPv6 address -	Instance state <a href="#">Running</a>	Public DNS -
Hostname type IP name: ip-10-0-1-84.ap-south-1.compute.internal	Private IP DNS name (IPv4 only) <a href="#">ip-10-0-1-84.ap-south-1.compute.internal</a>	Elastic IP addresses -
Answer private resource DNS name -	Instance type t2.micro	

## Step 6: Configure Security Group for EC2s

Allow SSH (22) from your IP and HTTP (80) from 0.0.0.0/0

## Step 7: Create a Target Group

Type: Instance, Protocol: HTTP, Port: 80, Health check path: /

The screenshot shows the AWS EC2 Target Groups page. On the left, there's a navigation sidebar with options like Volumes, Snapshots, Lifecycle Manager, Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces), Load Balancing (Load Balancers, Target Groups, Trust Stores), Auto Scaling (Auto Scaling Groups, Settings), and EC2 (Dashboard, Events, Instances, Images, Elastic Block Store). The main area displays a table titled 'Target groups (1/1)'. The table has columns for Name, ARN, Port, Protocol, and Target type. One row is selected, showing 'Web-Deployment-LB' with ARN 'arn:aws:elasticloadbalancing:ap-south-1:750311440127:targetgroup/Web-Deployment-LB/5dd69b899c59468a', Port '80', Protocol 'HTTP', and Target type 'Instance'. Below the table, a detailed view for 'Target group: Web-Deployment-LB' is shown with sections for Details (Target type: Instance, Protocol: Port HTTP: 80, Protocol version: HTTP1, VPC: vpc-01756043392ef4719), IP address type (IPv4, Load balancer: Web-Deployment-ALB), and a note about Anomaly mitigation: Not applicable.

The screenshot shows the AWS EC2 Registered Targets page. The left sidebar includes EC2 (Dashboard, Events, Instances, Images, Elastic Block Store), AMIs, and CloudShell. The main area shows a summary of registered targets: 2 Total targets, 2 Healthy, 0 Unhealthy, 0 Unused, 0 Initial, and 0 Draining. Below this is a table titled 'Registered targets (2)'. The table has columns for Instance ID, Name, Port, Zone, and Health status. Two rows are listed: 'i-073790655b83f4e7c' (web-deployment-1, 80, ap-south-1a, Healthy) and 'i-0ce4f75866ada822a' (web-deployment-2, 80, ap-south-1b, Healthy). A note at the top states: '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.'

## Step 8: Create an Application Load Balancer

Internet-facing, Listener on HTTP 80, Subnet-1 and Subnet-2, Attach target group

**Load balancers (1/1)**

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Name	Status	Type	Scheme	IP address type	VPC ID
Web-Deployment-ALB	Active	application	Internet-facing	IPv4	vpc-0175...

**Load balancer: Web-Deployment-ALB**

Load balancer type Application	Status <span style="color: green;">Active</span>	VPC <a href="#">vpc-01756043392ef4719</a>	Load balancer IP address type IPv4
Scheme Internet-facing	Hosted zone ZP97RAFLXTNZK	Availability Zones <a href="#">subnet-0299aa...</a> ap-south-1a (aps1-az1) <a href="#">subnet-0cdace...</a> ap-south-1b (aps1-az3)	Date created August 5, 2025, 17:36 (UTC+05:30)

**Web-Deployment-ALB**

**Details**

Load balancer type Application	Status <span style="color: green;">Active</span>	VPC <a href="#">vpc-01756043392ef4719</a>	Load balancer IP address type IPv4
Scheme Internet-facing	Hosted zone ZP97RAFLXTNZK	Availability Zones <a href="#">subnet-0299aa...</a> ap-south-1a (aps1-az1) <a href="#">subnet-0cdace...</a> ap-south-1b (aps1-az3)	Date created August 5, 2025, 17:36 (UTC+05:30)
Load balancer ARN <a href="#">arn:aws:elasticloadbalancing:ap-south-1:750311440127:loadbalancer/app/Web-Deployment-ALB/2aaf3b5bb37fb863</a>		DNS name info <a href="#">Web-Deployment-ALB-1839859024.ap-south-1.elb.amazonaws.com</a> (A Record)	

## Step 9: Test the ALB in Browser

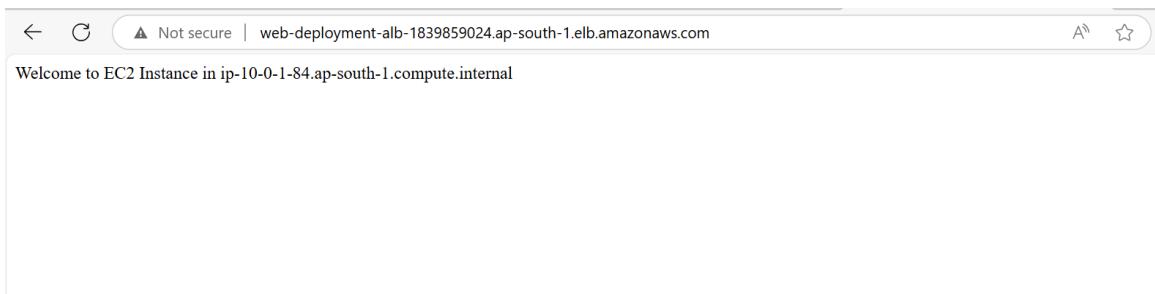
Find ALB DNS in EC2 → Load Balancers.

Open in browser: <http://<ALB-DNS-Name>>

[web-deployment-alb-1839859024.ap-south-1.elb.amazonaws.com](http://web-deployment-alb-1839859024.ap-south-1.elb.amazonaws.com)

Not secure | [web-deployment-alb-1839859024.ap-south-1.elb.amazonaws.com](http://web-deployment-alb-1839859024.ap-south-1.elb.amazonaws.com)

Welcome to EC2 Instance in ip-10-0-2-140.ap-south-1.compute.internal



Screenshot of the AWS EC2 Target Groups console for the 'Web-Deployment-LB' target group.

**Details:**

- arn:aws:elasticloadbalancing:ap-south-1:750311440127:targetgroup/Web-Deployment-LB/5dd69b899c59468a
- Target type: Instance
- Protocol: Port
- Protocol version: HTTP1
- VPC: vpc-01756043392ef4719
- IP address type: IPv4
- Load balancer: Web-Deployment-ALB

Total targets	Healthy	Unhealthy	Unused	Initial	Draining
2	2	0	0	0	0
0 Anomalous					

**Distribution of targets by Availability Zone (AZ):**  
Select values in this table to see corresponding filters applied to the Registered targets table below.

**Targets** | Monitoring | Health checks | Attributes | Tags

## Step 10: Test High Availability

Stop EC2-A. Refresh ALB DNS. Traffic should route to EC2-B.

Target group: Healthy: 1, Unused: 1

Screenshot of the AWS EC2 Instances console.

**Instances (1/2) Info:**

Name	Instance ID	Instance state	Instance type	Status check
web-deployment-server-2	i-0ce4f75866ada822a	Running	t2.micro	2/2 checks passed
web-deployment-server-1	i-073790655b83f4e7c	Stopped	t2.micro	-

**i-073790655b83f4e7c (web-deployment-server-1) Details:**

- Instance ID: i-073790655b83f4e7c
- Public IPv4 address: -
- Private IP4 addresses: 10.0.1.84
- Public DNS: -
- Instance state: Stopped
- Hostname type: IP name: ip-10-0-1-84.ap-south-1.compute.internal

Screenshot of a web browser showing the EC2 instance details. The URL is 'web-deployment-alb-1839859024.ap-south-1.elb.amazonaws.com'. The page title is 'Welcome to EC2 Instance in ip-10-0-2-140.ap-south-1.compute.internal'.

**Details**

Target type Instance	Protocol : Port HTTP: 80	Protocol version HTTP1	VPC <a href="#">vpc-01756043392ef4719</a>
IP address type IPv4	Load balancer <a href="#">Web-Deployment-ALB</a>		
2 Total targets	1 Healthy	0 Unhealthy	1 Unused
	0 Anomalous		0 Initial 0 Draining

**Distribution of targets by Availability Zone (AZ)**  
Select values in this table to see corresponding filters applied to the Registered targets table below.

**Targets** | Monitoring | Health checks | Attributes | Tags

## Routing Table Summary:

Destination: 0.0.0.0/0 → Target: Internet Gateway

## Final Output Verification:

Check	Result
ALB URL reachable	Yes
EC2 servers show web page	Yes
Stopping one EC2 works	Traffic goes to other EC2
Target group health status	One Healthy, One Unused

## Conclusion:

Successfully deployed a highly available web application in AWS using:

- Two EC2s in different AZs
- Application Load Balancer
- Proper routing tables and health checks

This setup ensures traffic is served even if one instance fails.