

## Practical - 8: Auto Scaling Group (ASG) with ALB

**Objective:** To configure a highly available, scalable infrastructure using a Custom AMI, Launch Template, Application Load Balancer (ALB), and an ASG with a dynamic scaling policy based on CPU utilization.

### Phase 1: Prepare the Custom AMI and Security Groups

#### Step 1: Create the Custom AMI (Base Image)

1. **Launch Base Instance:** Launch an EC2 instance ([pr8-ami-base](#)) using the **Amazon Linux 2023 AMI** ([t3.micro](#)).

**Install Software (User Data):** In the **Advanced details** section, use the following User Data script to install the Apache web server ([httpd](#)) and the load testing utility ([stress](#)).

Bash

```
#!/bin/bash
sudo dnf update -y
sudo dnf install -y httpd stress
sudo systemctl start httpd
sudo systemctl enable httpd
echo "<h1>Welcome to Practical 8 - ASG Test Server</h1>" > /var/www/html/index.html
```

2. **Create AMI:** Once the instance is running, select it, go to **Actions -> Image and templates -> Create image**.
  - **Image Name:** [pr8-custom-ami](#).

#### Step 2: Create Security Groups (SGs)

1. **Instance SG ([pr8-instance-sg](#)):** Create a security group for the EC2 instances.
  - **Name:** [pr8-instance-sg](#).
  - **Inbound Rules:** Allow SSH (My IP) and HTTP (from **Anywhere** initially).
2. **ALB SG ([pr8-alb-sg](#)):** Create a security group for the Application Load Balancer.
  - **Name:** [pr8-alb-sg](#).
  - **Inbound Rule:** Allow HTTP (Port 80) from **Anywhere** ([0.0.0.0/0](#)).
3. **Refine Instance SG (Security Best Practice):** Edit the inbound rules of [pr8-instance-sg](#) to ensure HTTP traffic on Port 80 is sourced **only** from the [pr8-alb-sg](#) Security Group ID.

## Phase 2: Create Load Balancer Components and Launch Template

### Step 3: Create Target Group and ALB

1. **Create Target Group ([practical8TG](#)):** Navigate to **Load Balancing -> Target Groups**.
  - **Name:** [practical8TG](#).
  - **Protocol/Port:** HTTP: 80.
2. **Create Application Load Balancer ([pr8-alb](#)):** Navigate to **Load Balancing -> Load Balancers**.
  - **Name:** [pr8-alb](#).
  - **Scheme:** Internet-facing.
  - **Mappings (AZs):** Select **at least two Availability Zones** (e.g., [ap-south-1a](#) and [ap-south-1b](#)).
  - **Security Groups:** Select [pr8-alb-sg](#).
  - **Listener/Routing:** Set the HTTP: 80 listener's Default Action to **Forward to the practical8TG Target Group**.

### Step 4: Create Launch Template

1. **Create Launch Template:** Navigate to **EC2 -> Launch Templates**.
  - **Name:** [pr8-Launch-Template](#).
  - **AMI:** Select the custom AMI, [pr8-custom-ami](#).
  - **Instance Type:** [t3.micro](#).
  - **Key Pair:** Select your key pair ([practical-key-pair](#)).
  - **Security Groups:** Select the existing [pr8-instance-sg](#).

## Phase 3: Create Auto Scaling Group (ASG) and Policy

### Step 5: Configure the ASG and Load Balancer Integration

1. **Create ASG:** Navigate to **Auto Scaling -> Auto Scaling Groups**.
  - **Group Name:** [pr8-asg](#).
  - **Launch Template:** Select [pr8-Launch-Template](#).
  - **Network:** Select the VPC and the **same two AZ subnets** used for the ALB.
2. **Integrate with Load Balancer:** Select **Attach to an existing load balancer**.
  - Select **Choose from your load balancer target groups** and select [practical8TG](#).

## Step 6: Define Capacity and Scaling Policy

1. **Group Size:** On the Configure group size and scaling step:
  - **Desired capacity:** 1.
  - **Minimum capacity:** 1
  - **Maximum capacity:** 3.
2. **Create Dynamic Scaling Policy:**
  - **Policy Type:** Target tracking scaling.
  - **Metric Type:** Average CPU utilization.
  - **Target Value:** 50 (The ASG will scale to maintain an average CPU utilization of 50%).

## Phase 4: Validation (Scale-Out/Scale-In Demonstration)

### Step 7: Scale-Out Test (Load Application)

1. **Verify Initial State:** Wait for the ASG to launch the first instance (Desired: 1, Min: 1, Max: 3) and confirm its status is **InService**.

**Apply Load:** SSH into the running instance and run the **stress** tool.

Bash

```
$ stress --cpu 1 --timeout 360
```

- **Observation:** The CPU utilization will breach the 50% target<sup>61</sup>. The policy will be triggered, changing the **Desired Capacity** from 1 to 2. The ASG launches a second instance. 62626262

### Step 8: Scale-In Test (Load Removal)

1. **Load Removal:** The **stress** command will automatically stop. CPU utilization will drop below the 50% target.
2. **Observation:** The policy will trigger a Scale-In event. The **Desired Capacity** will shrink from 2 to 1, and one instance will be **terminated**.
3. **Final Confirmation:** Review the ASG's **Activity History** to confirm the documented **Scale-Out** and **Scale-In** events.