Create ALB – ASG Demo using Cloudformation

1) What this template Do

It creates a small, working web service that automatically scales and sits behind a load balancer:

1. A VPC with two public subnets (so instances are reachable from the internet).
2. An Internet Gateway and routing so traffic can reach the subnets.
3. Security groups for the ALB and the EC2 instances.
4. An **Application Load Balancer (ALB)** that listens on HTTP port 80.
5. A **Target Group** for the ALB (where backend instances register).
6. A **Launch Template** defining how to start EC2 instances (AMI, type, SSH key, and startup script).
7. An **Auto Scaling Group (ASG)** that launches EC2 instances from the launch template, attaches them to the ALB target group, and maintains the desired number of healthy instances.
8. Output: the ALB's DNS name so you can open the site in your browser.

---

2) Step-by-step, plain-language flow

1. Networking: VPC + Internet Gateway + Subnets

- The template creates a VPC (a private network) with CIDR 10.0.0.0/16.
- It makes two public subnets (one in AZ A and one in AZ B). Having two AZs gives availability — if one AZ goes down, the other still serves traffic.
- It creates an Internet Gateway and a route (0.0.0.0/0 → Internet Gateway), and associates that route table with both subnets. That allows instances (and the ALB) in those subnets to talk to the internet.

2. Security Groups

- ALBSG: a security group that allows inbound HTTP (port 80) from anywhere (0.0.0.0/0) — this is applied to the ALB so the world can reach it.

- InstanceSG: allows inbound HTTP (port 80) from anywhere as well (in the demo). In a production setup you usually restrict InstanceSG to only allow traffic from the ALB SG.

3. Application Load Balancer (ALB) + Target Group + Listener

- ALB: a managed load balancer deployed into the two public subnets. It receives incoming HTTP requests.
- Target Group: a logical set of backend targets (in this template the targets are EC2 instances, TargetType: instance) on port 80.
- Listener: the ALB listens on port 80 and forwards incoming requests to the target group using the default action (forward).
- Health check: the ALB will periodically request / on each instance. If the response is HTTP 200, the instance is considered healthy and receives traffic.

4. Launch Template (how instances boot)

- This defines the instance type (t2.micro), the AMI (Amazon Linux 2 in the template), the SSH key pair (so you can SSH in), security group, and **User Data**.
- **User Data**: a short script that runs when the instance boots:
    - Installs and starts a web server (httpd).
    - Writes a simple HTML page showing the hostname.
    - This makes each launched instance return HTTP 200 on /, which satisfies the ALB health check.

5. Auto Scaling Group (ASG)

- ASG is set to MinSize: 2, DesiredCapacity: 2, MaxSize: 4. That means:
    - AWS will try to keep **2** instances running at all times.
    - If one instance fails its health check, ASG will terminate it and launch a replacement.
    - ASG can scale up to 4 instances if you later add scaling policies.
- ASG uses the Launch Template to create instances and automatically registers them with the ALB target group (because TargetGroupARNs is provided).

- HealthCheckType is ELB, so ASG uses the ALB health status to determine if instances are healthy.

6. Output: ALB DNS name

- CloudFormation prints the ALB DNS name in the stack outputs. Open that DNS in a browser and you'll see the HTML page served by whichever backend instance served your request.

---

3) How the pieces work together at runtime (traffic flow)

1. A user types the ALB DNS name into their browser.
2. DNS resolves to the ALB's public IPs in two AZs.
3. ALB listens on port 80 and receives the request.
4. ALB chooses a healthy target from the target group (based on health checks / round-robin).
5. ALB forwards the request to the chosen EC2 instance on port 80.
6. The instance's web server responds (the simple HTML page created by user-data).
7. ALB returns the response to the user.

Meanwhile:

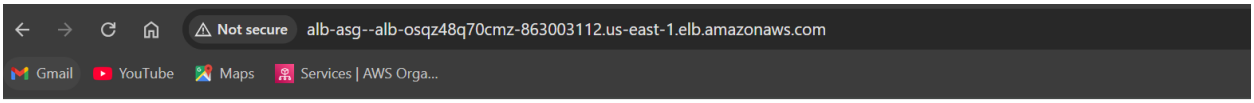
- ASG monitors instance health via the ALB. If an instance is unhealthy, ASG replaces it automatically.
- If you add load-based scaling policies later, ASG can increase/decrease instance count between Min and Max.

Important practical notes & tips

- **AMI is region-specific.** The provided AMI (ami-0c02fb55956c7d316) is Amazon Linux 2 in one region. If you deploy in a different region, the AMI ID may not exist there. Replace the ImageId with a region-appropriate Amazon Linux 2 AMI or use an SSM parameter to fetch the latest AMI.
- **KeyPairName**: you must pass an existing EC2 key pair name (the template has a parameter). If you don't need SSH access, you can remove the KeyPair requirement.

- **Security**: The example allows HTTP from everywhere to the instances. For better security, restrict InstanceSG to only allow traffic from ALBSG.
- **Costs**: ALB + EC2 instances may incur costs even on Free Tier (ALB not free). Clean up (delete the stack) when done to avoid charges.
- **Scaling policies**: This template has no dynamic scaling configured. You can add CloudWatch alarms and ASG scaling policies to scale on CPU or request count.
- **Health checks path**: User data writes content at /. If your app serves at a different path, change the ALB HealthCheckPath accordingly.

Output



Welcome to Auto Scaling + ALB Demo from ip-10-0-2-188.ec2.internal