**AWS Auto Scaling Groups - Complete Step-by-Step Guide with SNS & Dynamic Scaling**

**Table of Contents**

1. Introduction to Auto Scaling
2. Vertical vs Horizontal Scaling
3. Lab Architecture
4. Step-by-Step Implementation
   * Step 1: Create Launch Template
   * Step 2: Set Up Target Group
   * Step 3: Configure Application Load Balancer
   * Step 4: Route 53 DNS Configuration
   * Step 5: SNS Notifications Setup
   * Step 6: Create Auto Scaling Group
   * Step 7: Configure Dynamic Scaling Policies
5. Testing & Validation
6. Cleanup
7. Real-World Use Cases

**1. Introduction to Auto Scaling**

Auto Scaling automatically adjusts EC2 capacity based on:

* **Traffic demand** (scale-out during peaks)
* **Cost optimization** (scale-in during low usage)
* **High availability** (replace unhealthy instances)

**Key Components Used:**

* **Launch Template** (AMI with pre-configured NGINX)
* **Target Groups & ALB** (Traffic distribution)
* **Route 53** (DNS routing)
* **SNS** (Alerts for scaling events)
* **CloudWatch Alarms** (Trigger scaling policies)

**2. Vertical vs Horizontal Scaling**

| **Feature** | **Vertical Scaling** | **Horizontal Scaling** |
| --- | --- | --- |
| **Approach** | Increase instance size (e.g., t2.micro → t2.large) | Add more instances |
| **Downtime** | Requires instance restart | Zero downtime |
| **Use Case** | Monolithic apps | Distributed systems (e.g., Netflix) |
| **AWS Service** | Manual resizing | Auto Scaling Groups |

**3. Lab Architecture**

A diagram of a software system

AI-generated content may be incorrect.

[User] -->|www.cloudvishwakarma.in| B[Route 53]

**4. Step-by-Step Implementation**

**Step 1: Create Launch Template**

1. **EC2 → Launch Templates → Create**
   * **Name:** asg-launch-template
   * **AMI:** Custom AMI with NGINX (pre-built via Packer)
   * **Instance Type:** t2.large
   * **User Data:**

bash

#!/bin/bash

sudo apt-get update

sudo apt-get install -y ansible

ansible-pull -U https://github.com/your-repo/nginx-config.git playbook.yml

**Step 2: Set Up Target Group**

1. **EC2 → Target Groups → Create**
   * **Name:** asg-target-group
   * **Protocol:** HTTP
   * **Port:** 80
   * **Health Check Path:** /

**Step 3: Configure Application Load Balancer**

1. **EC2 → Load Balancers → Create ALB**
   * **Name:** asg-alb
   * **Scheme:** Internet-facing
   * **Listeners:**
     + HTTP/80 → Redirect to HTTPS/443
     + HTTPS/443 → Forward to asg-target-group
   * **Public Subnets:** Select across 3 AZs

**Step 4: Route 53 DNS Configuration**

1. **Route 53 → Hosted Zones → Create Record**
   * **Name:** www.cloudvishwakarma.in
   * **Alias:** Yes → Point to asg-alb

**Step 5: SNS Notifications Setup**

1. **SNS → Topics → Create Topic**
   * **Name:** asg-sns-alerts
2. **Subscribe Email** (Confirm via email link)

**Step 6: Create Auto Scaling Group**

1. **EC2 → Auto Scaling Groups → Create**
   * **Launch Template:** asg-launch-template
   * **VPC & Subnets:** Select 3 public subnets
   * **Load Balancing:** Attach asg-alb
   * **Group Size:**
     + Min: 1
     + Desired: 1
     + Max: 4
   * **Notifications:** Add SNS topic

**Step 7: Configure Dynamic Scaling Policies**

**Scale-Out Policy (High CPU)**

1. **CloudWatch → Alarms → Create**
   * **Metric:** CPUUtilization (Maximum, 1 minute)
   * **Threshold:** > 40% for 1 minute
   * **Actions:**
     + Notify SNS
     + Add 1 instance (Step Scaling)
2. **Repeat for 60% (Add +1) and 80% (Add +1)**

**Scale-In Policy (Low CPU)**

1. **CloudWatch → Alarms → Create**
   * **Metric:** CPUUtilization (Maximum, 1 minute)
   * **Threshold:** < 35% for 5 minutes
   * **Actions:**
     + Notify SNS
     + Remove 1 instance

**5. Testing & Validation**

1. **Generate CPU Load:**

bash

stress --cpu 2 --timeout 900 *# Simulate 100% CPU for 15 mins*

1. **Verify Scaling:**
   * Check **EC2 Console** for new instances
   * Monitor **SNS Email Alerts**
2. **Test ALB Routing:**

bash

watch -n 1 'curl -s http://www.cloudvishwakarma.in | grep "Server"'

**6. Cleanup**

1. Delete **Auto Scaling Group**
2. Terminate **EC2 Instances**
3. Delete **ALB & Target Groups**
4. Remove **Route 53 Record**
5. Delete **SNS Topic**

**7. Real-World Use Cases**

| **Scenario** | **Auto Scaling Benefit** |
| --- | --- |
| **E-commerce Sale** | Handles 10x traffic spikes |
| **Banking App** | Zero downtime during maintenance |
| **Streaming Service** | Scales globally for peak hours |

🚀 **Pro Tip:** Combine with **AWS WAF** for security during scaling events!  
📧 **Next:** Learn how to integrate **Lambda for serverless scaling triggers**.