This lab demonstrates the steps from *Demo: Auto Scaling*.
My full AWS Architect Associate course can be found here:
https://www.udemy.com/course/ultimateaws/?referralCode=7ED214B795C444141361

Lab Guide: Setting Up an Auto Scaling Group for EC2 in AWS

This lab guide walks you through the process of setting up an **Auto Scaling Group (ASG)** for EC2 instances in AWS. By following along, you will understand how to create a **Launch Template**, configure the **Auto Scaling Group**, and test the automatic scaling of EC2 instances.

Lab Objectives

1. Create a **Launch Template** as a blueprint for EC2 instances.
2. Configure an **Auto Scaling Group** to manage instances dynamically.
3. Understand how scaling policies adjust the number of instances based on demand.

Step 1: Access the AWS Console

1. Log in to the **AWS Management Console** and ensure you are in the **North Virginia (us-east-1)** region.
2. Navigate to the **EC2 Dashboard** by clicking on **EC2** from the AWS home screen.

Step 2: Create a Launch Template

2.1 Start the Launch Template Creation

1. From the left-hand menu, select **Launch Templates** under the "Instances" section.
2. Click **Create Launch Template**.

2.2 Configure the Launch Template

1. **Name**: Enter a name for the template, such as `DemoLT`.
2. **Template Version**: Leave as version 1 (default).
3. **AMI (Amazon Machine Image)**: Choose **Amazon Linux 2 AMI**.
4. **Instance Type**: Select **t2.micro** (free tier eligible).
5. **Key Pair**: Choose an existing key pair or create a new one to allow SSH access.
6. **Network Settings**:
    - Do not specify a subnet (the ASG will handle this).
    - Select an existing **security group**. For this example, choose a security group that allows:
        - **SSH** (port 22) for access.
        - **HTTP** (port 80) for web traffic.
7. **User Data (Optional)**: Add a script to automatically configure instances:

```
#!/bin/bash
sudo yum update -y
sudo yum install -y httpd
sudo systemctl start httpd
sudo systemctl enable httpd
echo "Welcome to Auto Scaling!" > /var/www/html/index.html
```

8. Scroll to the bottom and click **Create Launch Template**.

Step 3: Create an Auto Scaling Group

3.1 Start the Auto Scaling Group Setup

1. Go back to the **EC2 Dashboard**.
2. From the left-hand menu, click on **Auto Scaling Groups** under the "Auto Scaling" section.
3. Click **Create Auto Scaling Group**.

3.2 Configure the Auto Scaling Group

1. **Name**: Enter a name, such as `DemoASG`.
2. **Launch Template**: Select the Launch Template you just created (`DemoLT`).
3. Click **Next**.

3.3 Define Network Settings

1. **VPC**: Choose the default VPC or another VPC if configured.
2. **Availability Zones**:
    - Select multiple Availability Zones (e.g., us-east-1a and us-east-1b) to ensure high availability.
3. Click **Next**.

3.4 Configure Load Balancing (Optional)

1. **Load Balancer**: For simplicity, do not associate a load balancer in this demo. In production, you would typically use an Elastic Load Balancer (ELB) to distribute traffic across instances.
2. Click **Next**.

3.5 Set Desired Capacity and Scaling Policies

1. **Group Size**:
    - **Desired Capacity**: 2 (start with 2 instances).
    - **Minimum Capacity**: 1 (ensure at least 1 instance is always running).
    - **Maximum Capacity**: 5 (allow up to 5 instances).
2. **Scaling Policy**:
    - Select **Target Tracking Scaling Policy**.
    - Configure the target value for **CPU Utilization**:
        - Example: Maintain an average of **70% CPU utilization** across instances.
    - Auto Scaling will add instances when CPU utilization exceeds 70% and remove instances when it falls below 70%.
3. Click **Next**.

3.6 Review and Create

1. Review all configurations, then click **Create Auto Scaling Group**.

Step 4: Monitor the Auto Scaling Group

4.1 View Instances

1. In the Auto Scaling Group dashboard, click on your newly created ASG (`DemoASG`).
2. Go to the **Instance Management** tab.
3. You will see two EC2 instances being created (as per the desired capacity).

4.2 Test Automatic Scaling

1. Simulate high CPU usage to trigger scaling out (launching more instances).
    - For simplicity, manual testing or more advanced workload generators can be used in production setups.
2. Observe how Auto Scaling adds or removes instances based on the scaling policy.

Step 5: Key Concepts Explained

5.1 Launch Template

- A **Launch Template** is a reusable blueprint defining how EC2 instances should be configured.
- Key elements include:
    - AMI (operating system)
    - Instance type (e.g., t2.micro)
    - Key pair, security groups, and user data.

5.2 Auto Scaling Group

- An **Auto Scaling Group (ASG)** ensures the correct number of EC2 instances are running to handle workload demands.
- It uses the Launch Template to create instances and adjusts their number dynamically based on scaling policies.

5.3 Scaling Policies

- **Target Tracking Scaling**: Automatically adjusts capacity to maintain a target metric (e.g., 70% CPU utilization).
- **Scheduled Scaling**: Adjusts capacity based on pre-defined schedules.
- **Dynamic Scaling**: Adds or removes instances based on custom policies triggered by CloudWatch alarms.

Step 6: Clean Up

1. To avoid incurring costs, delete resources created during the lab:
    - Terminate all EC2 instances in the ASG.
    - Delete the Auto Scaling Group.
    - Delete the Launch Template.

Summary

In this lab, you learned how to configure an Auto Scaling Group to dynamically manage EC2 instances based on demand. You created a Launch Template as a blueprint for the instances and used a Target Tracking Scaling Policy to maintain optimal performance and cost efficiency. By automating scaling, AWS Auto Scaling ensures that your applications remain highly available while minimizing resource usage and costs.