**Placement Empowerment Program**

***Cloud Computing and DevOps Centre***

**Implement Auto-scaling in the Cloud**Set up an auto-scaling group for your cloud VMs to handle variable workloads.

Name: Mathimalar.P Department : ADS



Introduction and Overview

Auto-scaling in the cloud enables virtual machines (VMs) to automatically adjust based on workload demands. It ensures that resources scale up during high traffic and scale down when demand decreases, optimizing performance and cost. Cloud platforms like AWS, Azure, and Google Cloud provide auto-scaling capabilities to maintain system stability and efficiency without manual intervention.

Objective

.

The goal is to set up an auto-scaling group to manage cloud VMs dynamically based on workload variations. This includes:

* Defining scaling policies to automatically add or remove instances.
* Setting thresholds based on CPU utilization, network traffic, or other performance metrics.
* Configuring load balancers to distribute traffic efficiently.
* Ensuring high availability and cost optimization by adjusting resources dynamically.

.

Importance

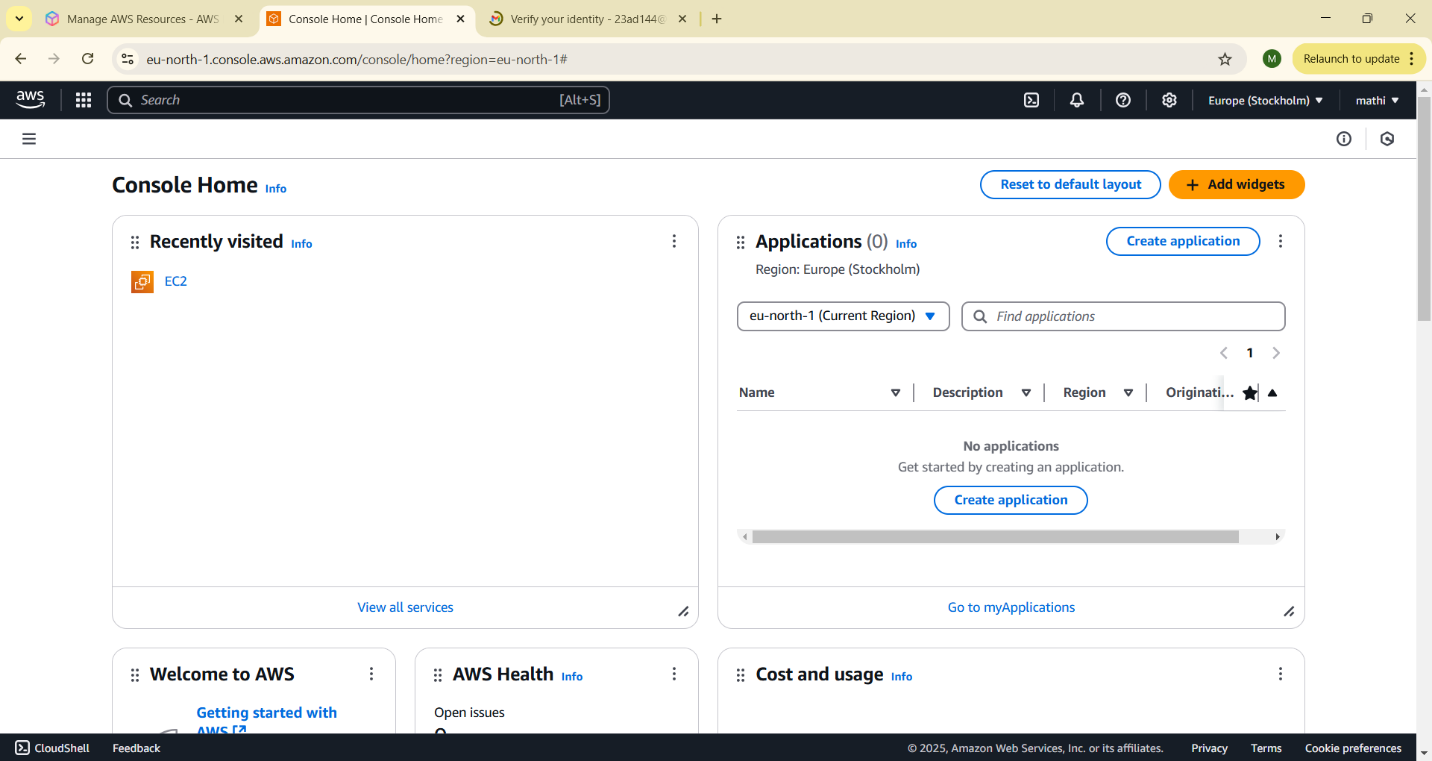
* **Optimized Performance** – Prevents service slowdowns by scaling up resources when needed.
* **Cost Efficiency** – Reduces unnecessary expenses by scaling down during low demand.
* **High Availability** – Ensures continuous service operation even during peak loads.
* **Automated Resource Management** – Reduces the need for manual scaling efforts.
* **Improved Fault Tolerance** – Replaces failed instances automatically, maintaining reliability.

By implementing auto-scaling, cloud environments can efficiently handle fluctuating workloads while ensuring stability and cost-effectiveness.

**Step-by-Step Overview**

**1. Set Up an AWS Account**

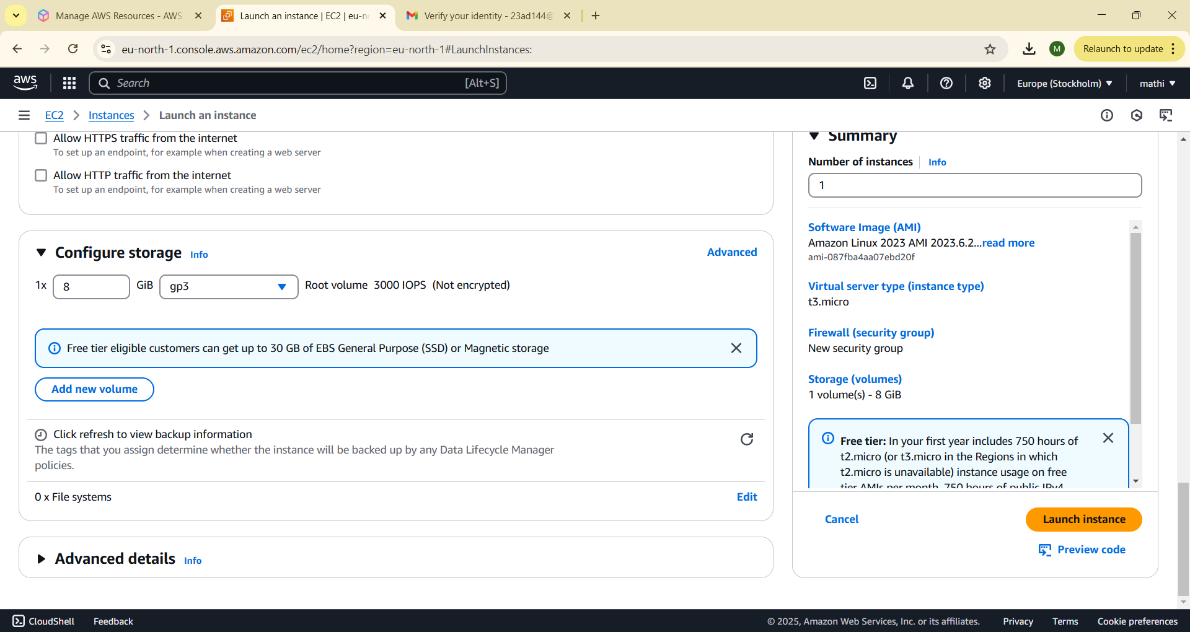
* If you don’t have an AWS account, create one at [AWS Console](https://aws.amazon.com/).
* Sign in to the **AWS Management Console**.



**2. Create a Launch Template**

(A launch template defines the settings for your Virtual Machines (EC2 instances)).

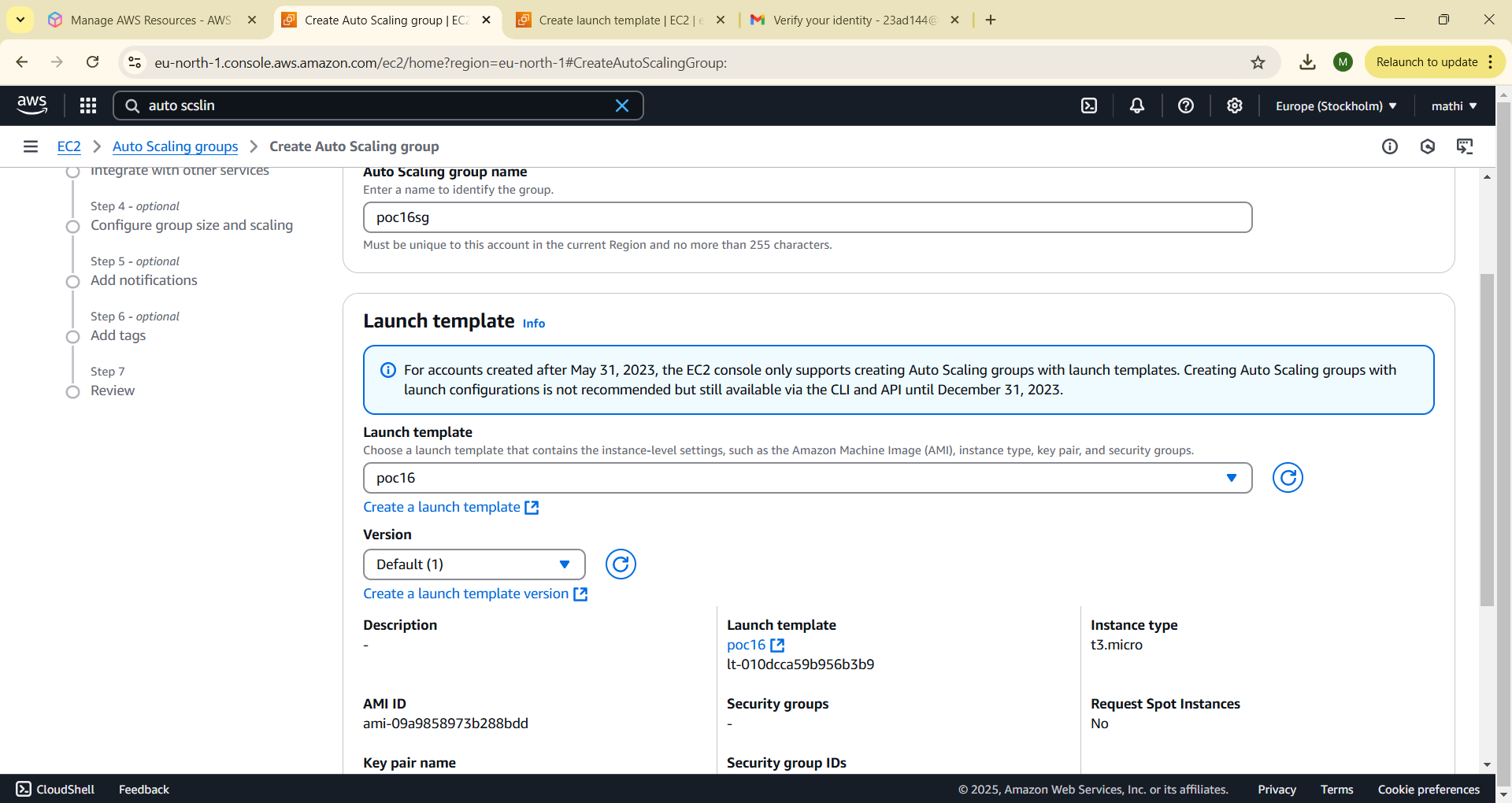
1. Open **EC2 Dashboard** in AWS Console.
2. Click on **Launch Templates** (left panel).
3. Click **Create Launch Template**.
4. Provide a **name** for the template.
5. Choose an **Amazon Machine Image (AMI)** (e.g., Amazon Linux 2).
6. Select an **instance type** (e.g., t2.micro).
7. Set **key pair** (for SSH access).
8. Configure **security group** (allow SSH, HTTP, etc.).
9. Click **Create Launch Template**.



**3. Create an Auto Scaling Group**

(An Auto Scaling Group (ASG) manages multiple instances to handle traffic changes).

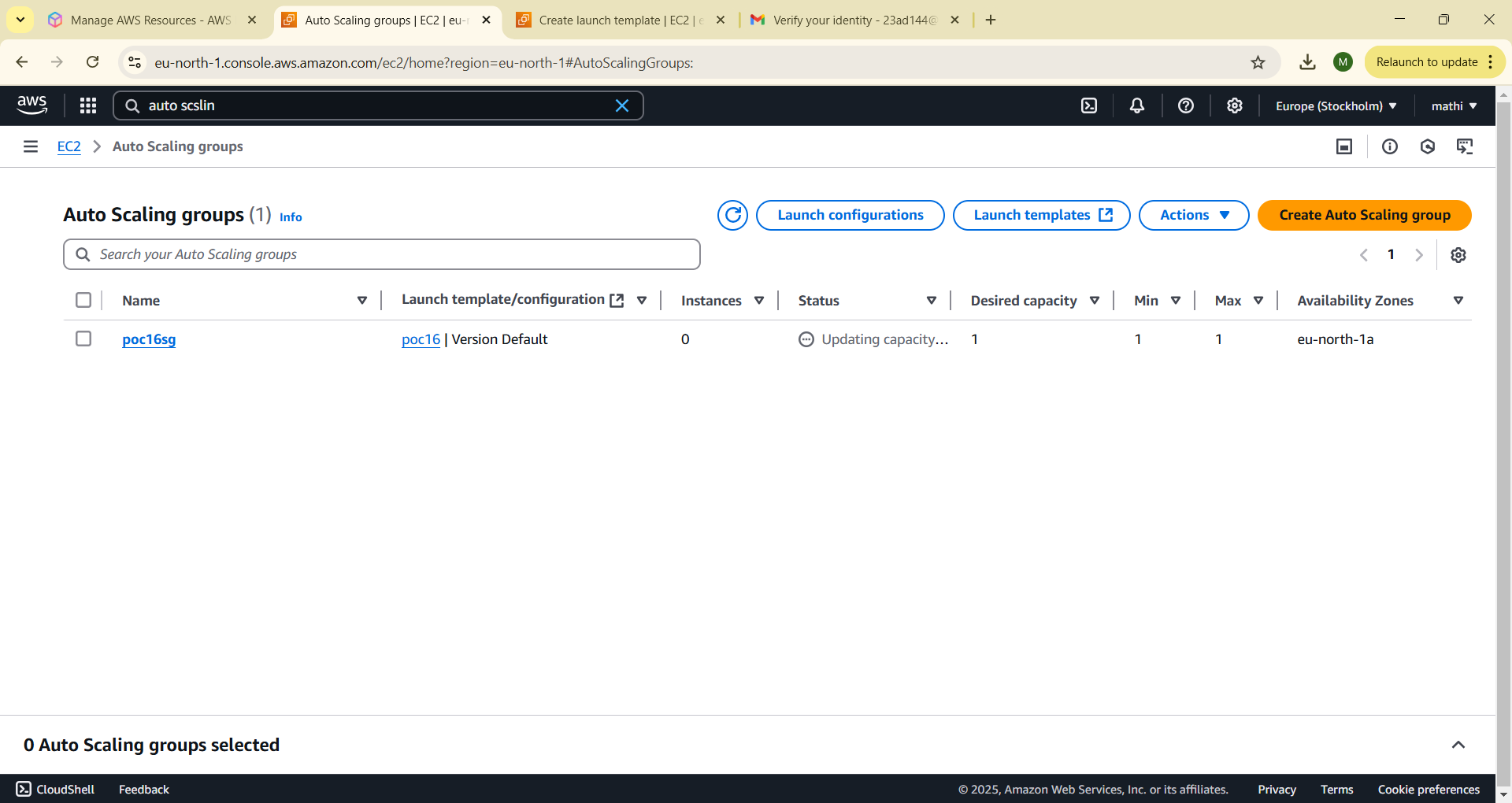
1. Go to the **Auto Scaling Groups** section in the AWS Console.
2. Click **Create Auto Scaling Group**.
3. Enter a **name** for the group.
4. Select the **Launch Template** created earlier.
5. Choose the **VPC and subnets** (to define where instances will run).
6. Click **Next** to proceed.



**4. Define Scaling Policies**

(Scaling policies control when to add or remove instances).

1. Choose **Target Tracking Scaling**.
2. Select **CPU Utilization** as the metric.
3. Set a **target value** (e.g., scale up when CPU > 70%).
4. Configure the **minimum and maximum number of instances** (e.g., Min = 1, Max = 5).
5. Click **Create Auto Scaling Group**.



**5. Test Auto-Scaling**

(Simulate high CPU usage to trigger auto-scaling).

1. Connect to an EC2 instance using SSH.

css

CopyEdit

ssh -i your-key.pem ec2-user@your-instance-ip

1. Run a high CPU load command:

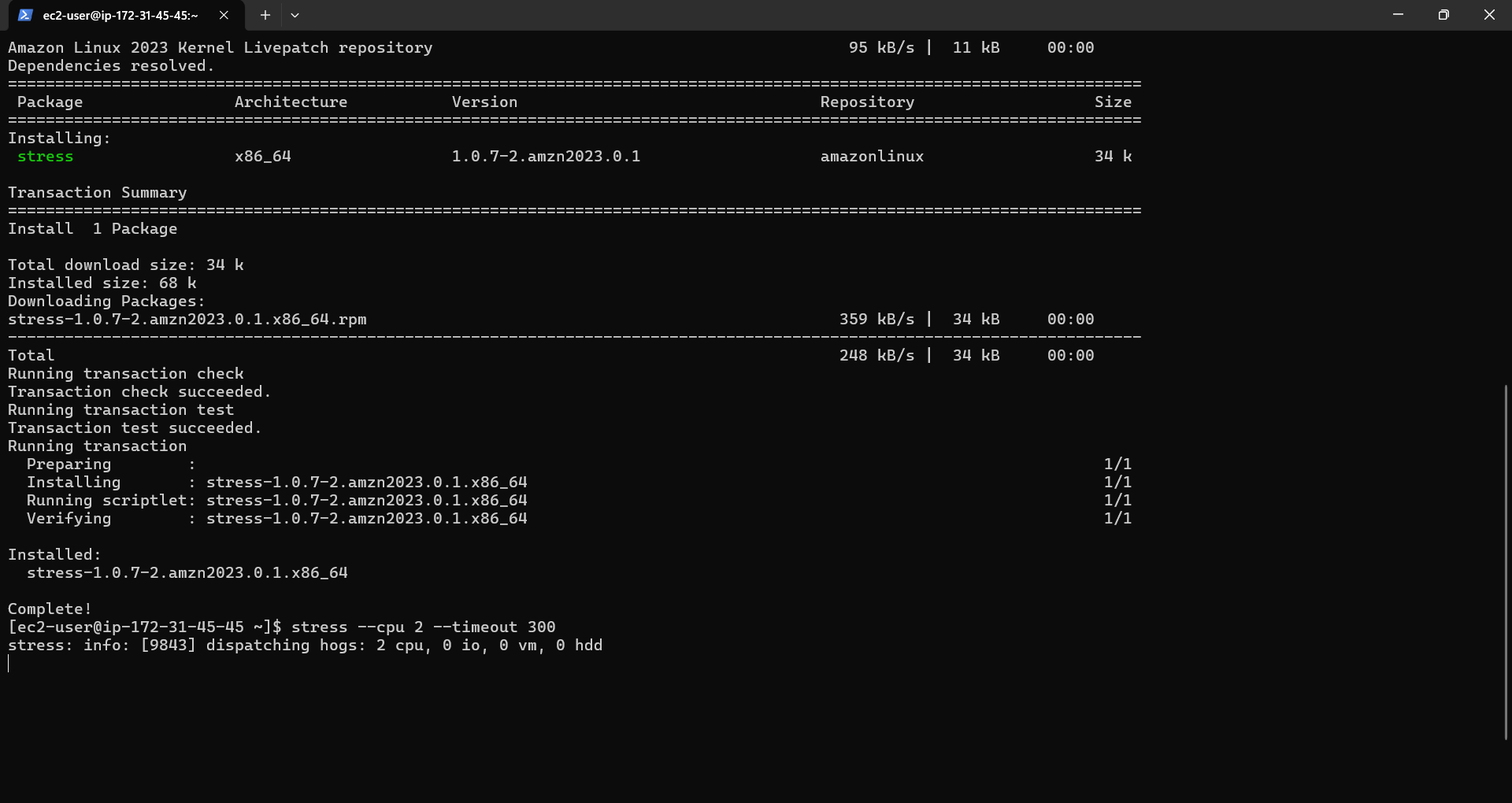
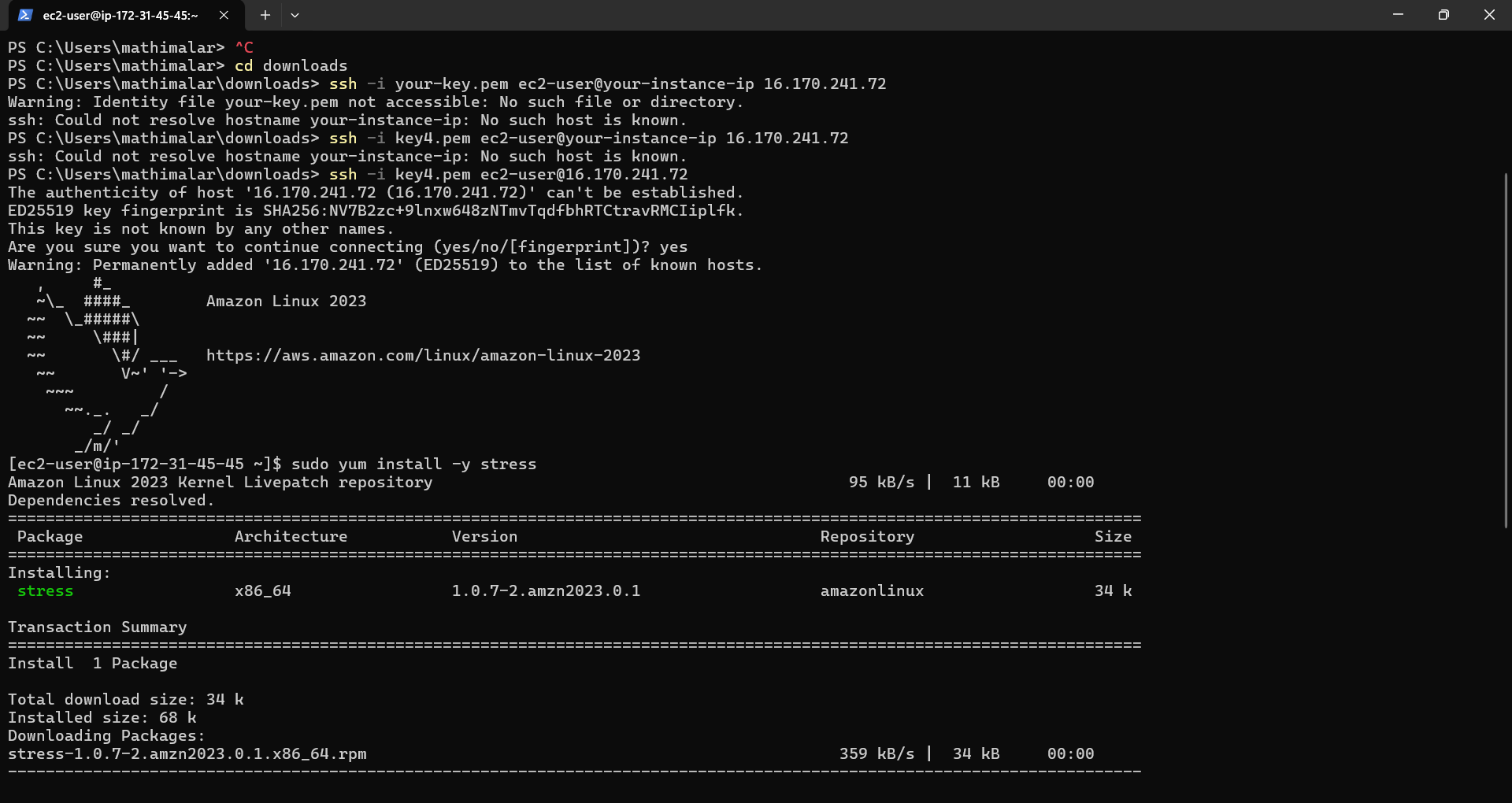
css

CopyEdit

sudo yum install -y stress

stress --cpu 2 --timeout 300

1. Check the **Auto Scaling Group** in AWS Console.
   * If CPU usage crosses 70%, new instances should launch.
   * When CPU usage drops, instances should terminate automatically.



**6. Monitor and Adjust**

* Go to **CloudWatch** in AWS Console to monitor auto-scaling activity.
* Adjust the **scaling policies** based on performance needs.

**Done! 🎉**

You have successfully set up Auto-Scaling for your cloud VMs! 🚀

**Expected Outcome**

* Auto-scaling in the cloud provides:
* **Optimized Resource Use**: Automatically adjusts VMs based on demand.
* **Cost Efficiency**: Scales resources up or down to match workload, saving costs.
* **High Availability**: Ensures your application stays online, even if instances fail.
* **Improved Performance**: Handles traffic spikes without slowdowns.
* **Fault Tolerance**: Automatically replaces failed instances.
* **Flexibility**: Scales resources elastically as traffic changes.
* In short, auto-scaling ensures better performance, cost savings, and reliability.