Thanks for dropping Task 1 and 2, Muhammad — this is solid material. Let’s format it into a clean, GitHub-ready Word doc structure that reflects your strategic clarity and modular thinking. I’ll also flag exactly which screenshots to take and where to place them.

**🧪 AZ-104 Lab 08: Manage Virtual Machines**

**Author:** Muhammad  
**Date Completed:** [Insert Date]  
**Environment:** Azure Portal (Campus Lab)  
**Security Note:** Screenshots only — no screen recordings or exposed credentials.

**✅ Lab Overview**

This lab explores the deployment and scaling of Azure Virtual Machines (VMs) and Virtual Machine Scale Sets (VMSS).  
You’ll manually deploy zone-resilient VMs, scale compute and storage resources, and prepare for autoscaling via VMSS.

**🧩 Task 1: Deploy Zone-Resilient Azure Virtual Machines**

**Objective:**  
Deploy two virtual machines across different availability zones to achieve a 99.99% uptime SLA.

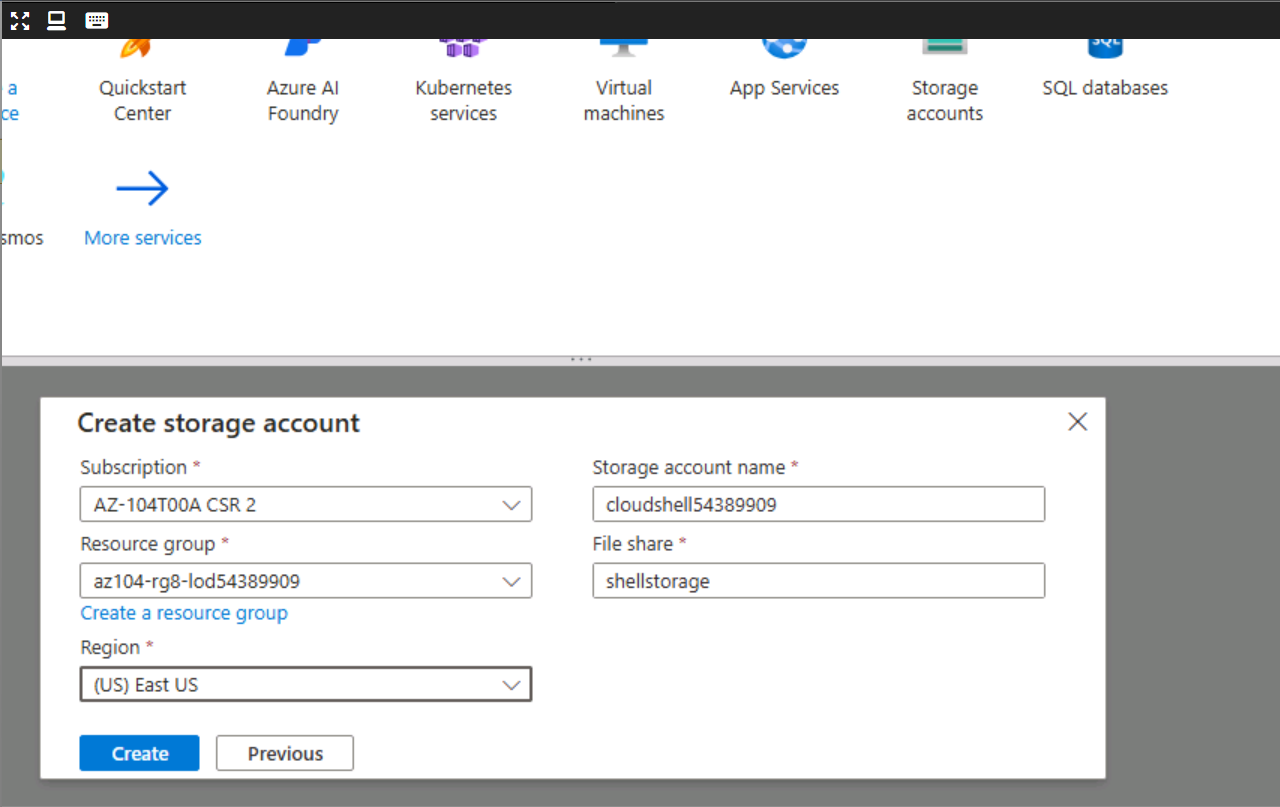
**Steps Taken:**

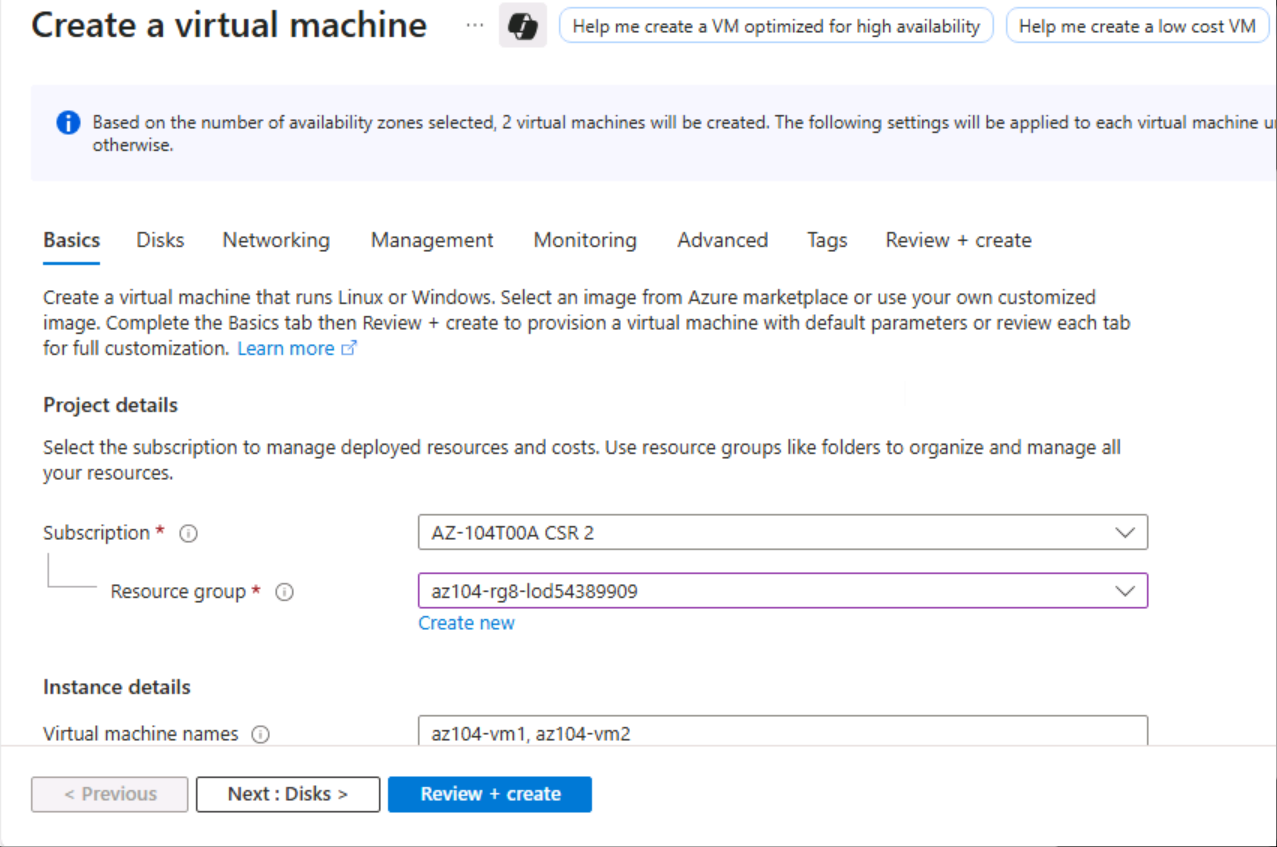
1. Signed into [Azure Portal](https://portal.azure.com)
2. Navigated to **Virtual Machines > + Create > Azure Virtual Machine**
3. Selected **Zone 1 and Zone 2** under Availability Zone
4. Configured the following settings:

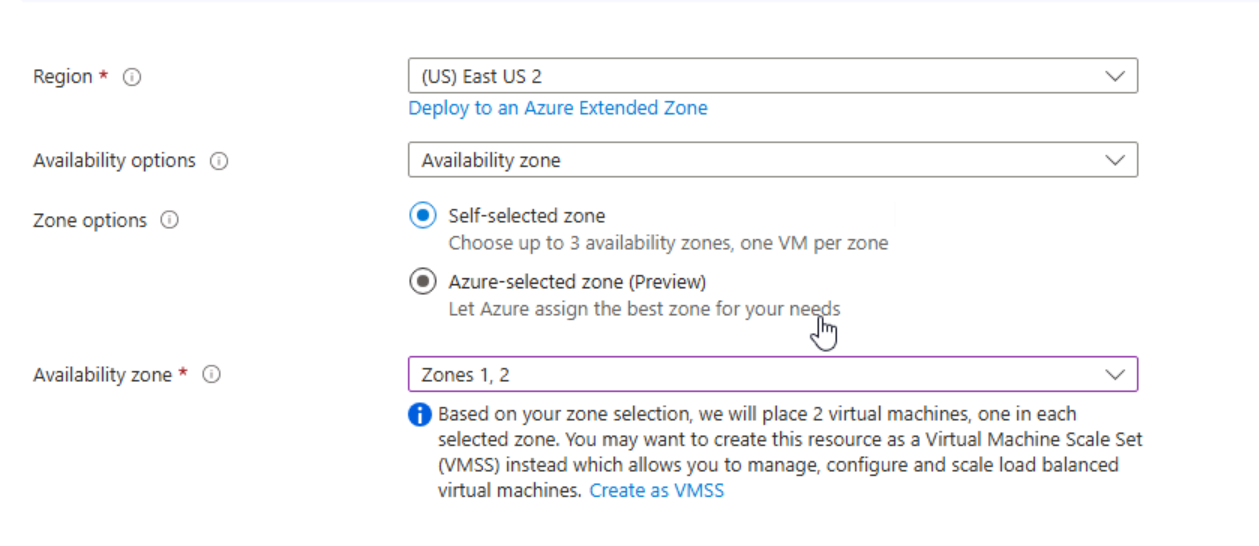
| **Setting** | **Value** |
| --- | --- |
| Resource Group | az104-rg8 |
| VM Names | az104-vm1, az104-vm2 |
| Region | East US |
| Image | Windows Server 2019 Datacenter - x64 Gen2 |
| Size | Standard D2s v3 |
| Username | localadmin |
| Password | [Secure] |
| Public Inbound Ports | None |
| OS Disk Type | Premium SSD |
| NIC & Public IP | Delete on VM deletion |
| Load Balancer | None |
| Patch Orchestration | Azure orchestrated |
| Boot Diagnostics | Disabled |

**Screenshot Checklist:**

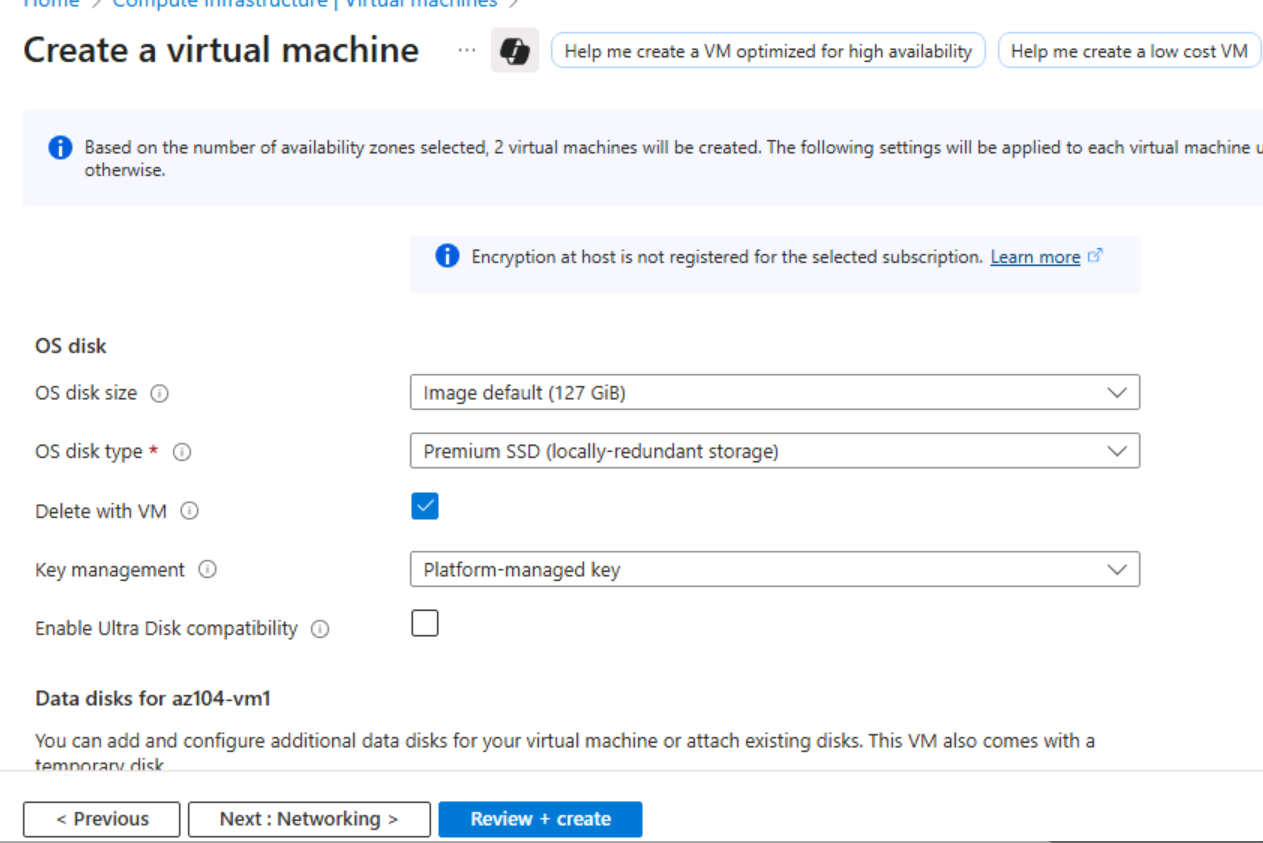
**1. created a storage account using the cloud shell**

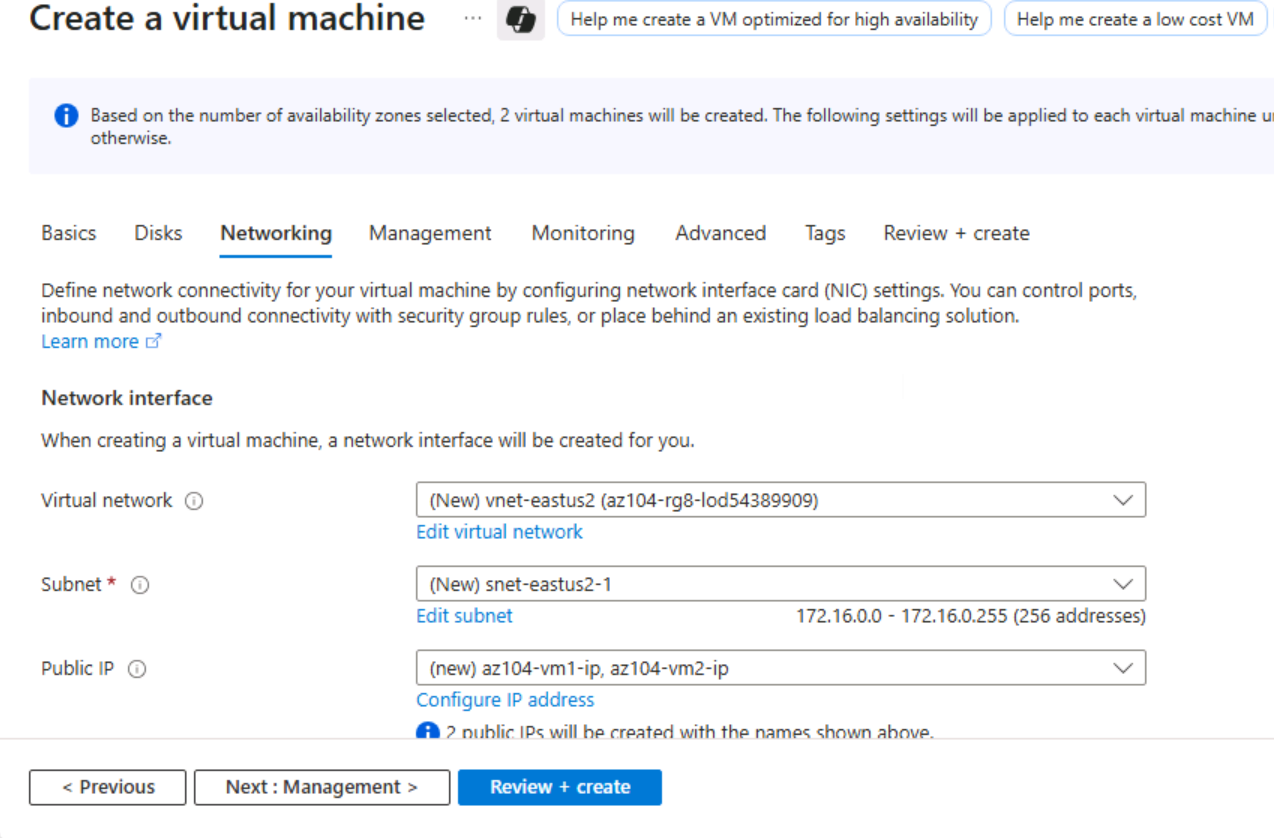
  
✅ VM creation page with both zones selected

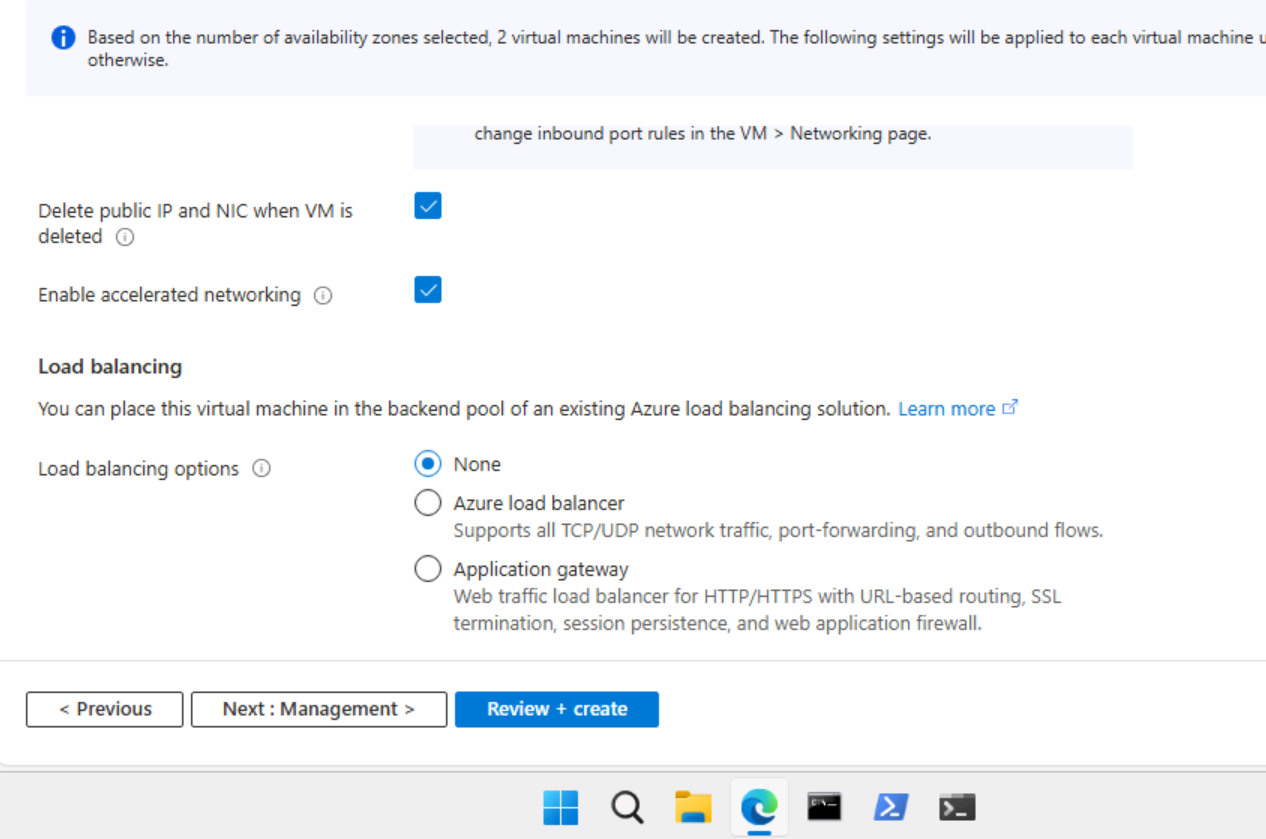


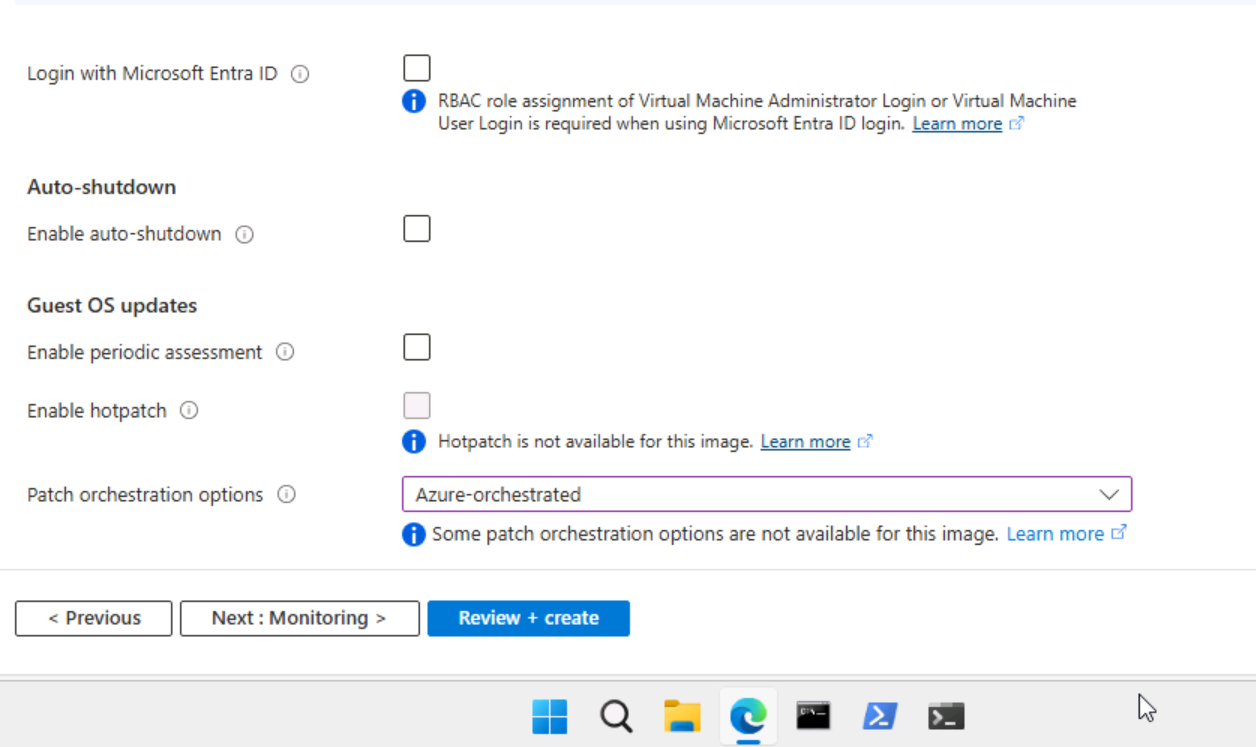


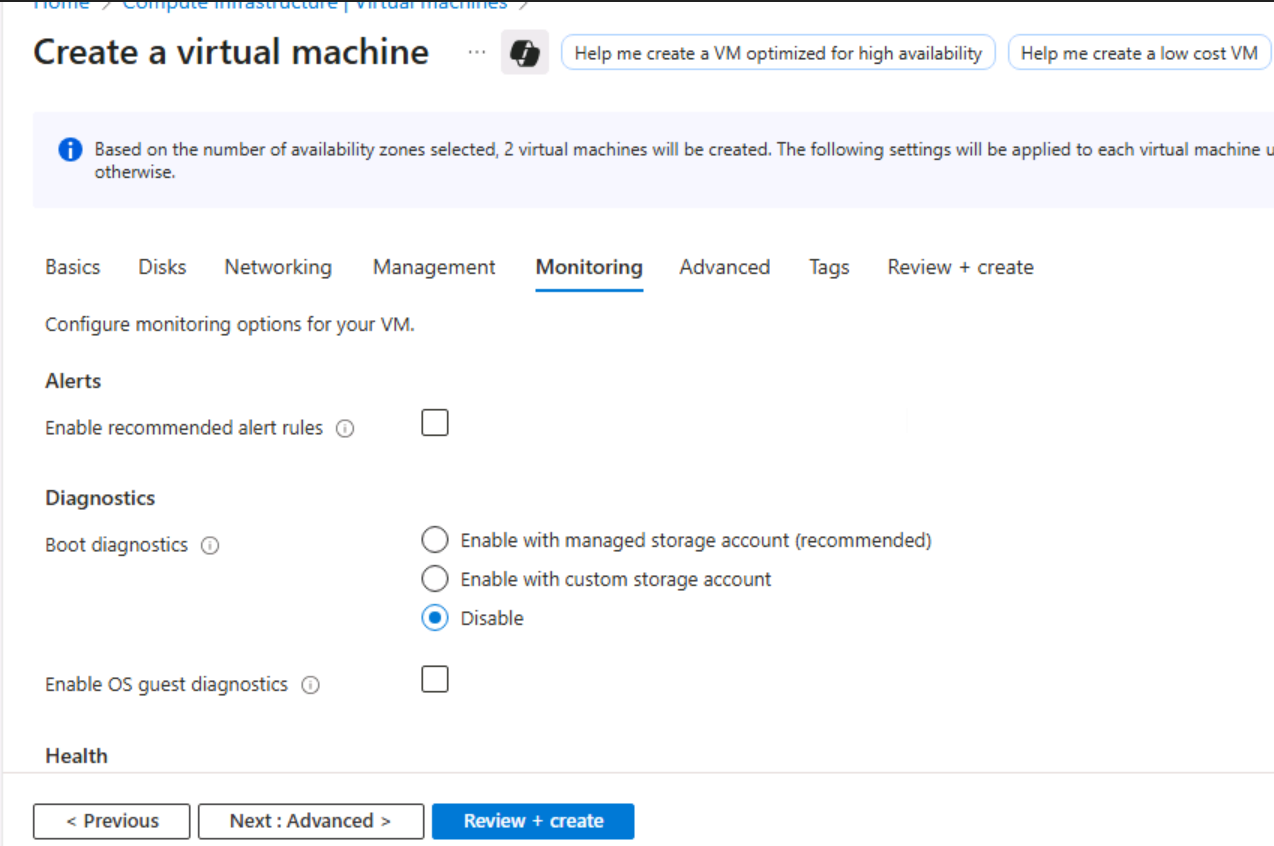
✅ Disk configuration (Premium SSD)

  
✅ Networking tab showing no load balancer

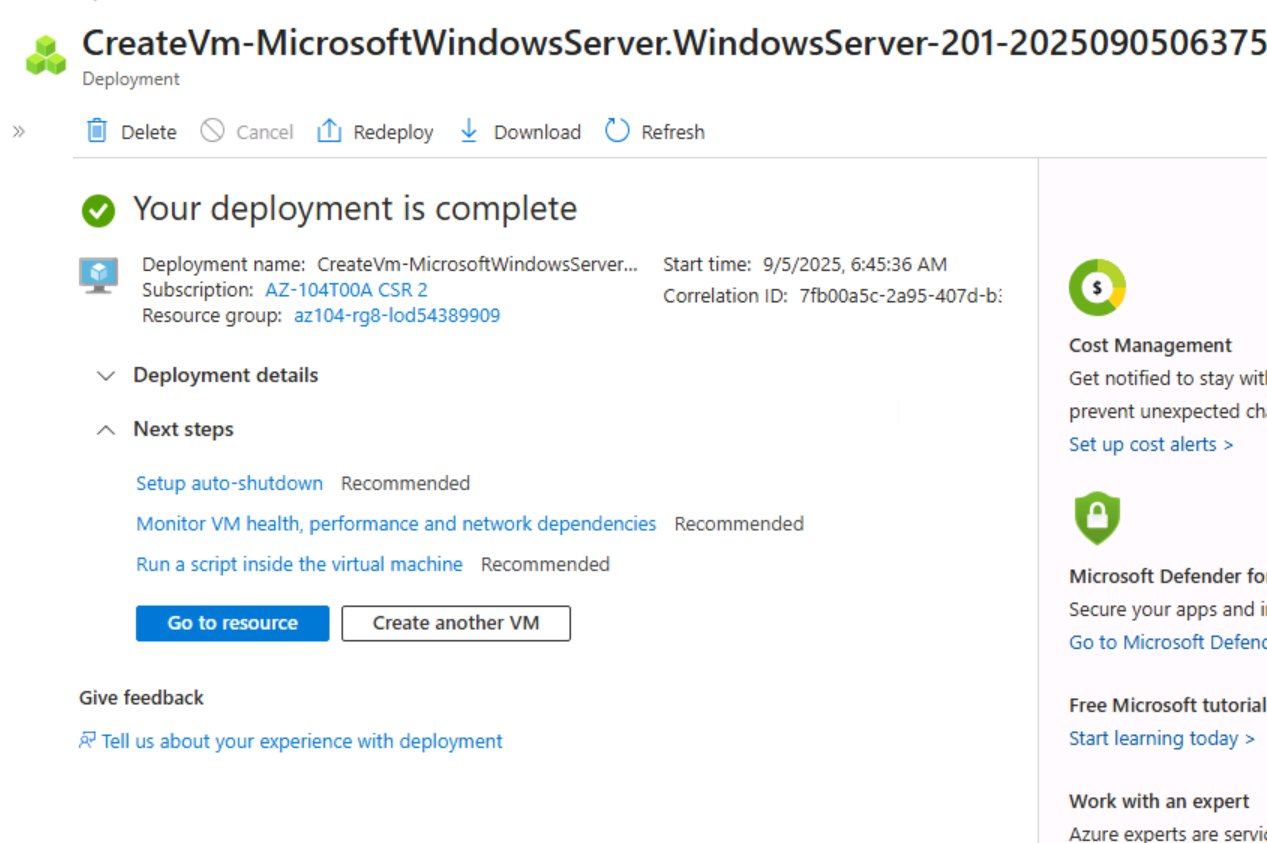


  
✅ Management tab with patch orchestration

  
✅ Monitoring tab with Boot Diagnostics disabled

  
✅ Final Review + Create screen

✅ Deployment complete notification



**Notes:**

* VM deployment automatically created NICs, disks, and public IPs as separate resources
* Availability Zones ensure high SLA and fault tolerance

**🧩 Task 2: Manage Compute and Storage Scaling for Virtual Machines**

**Objective:**  
Vertically scale VM compute resources and modify attached disk performance.

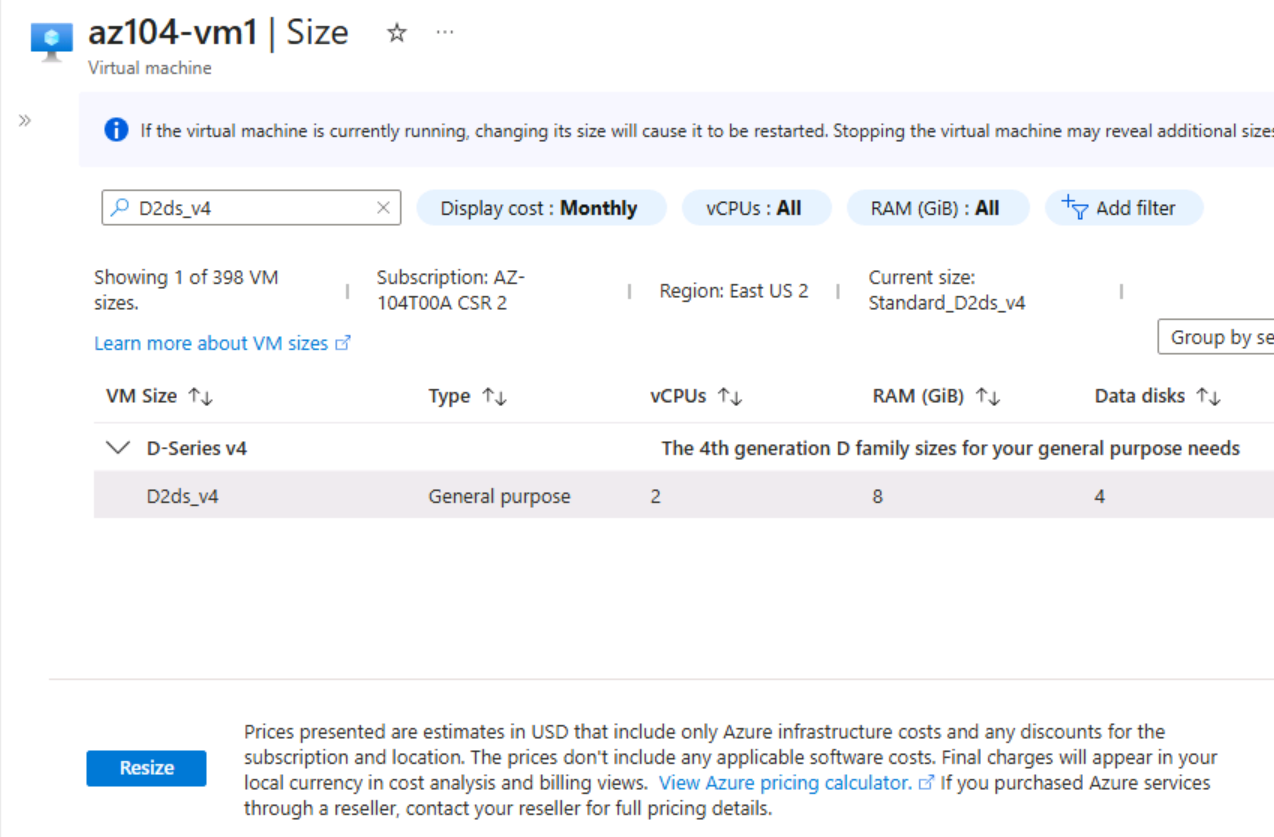
**Steps Taken:**

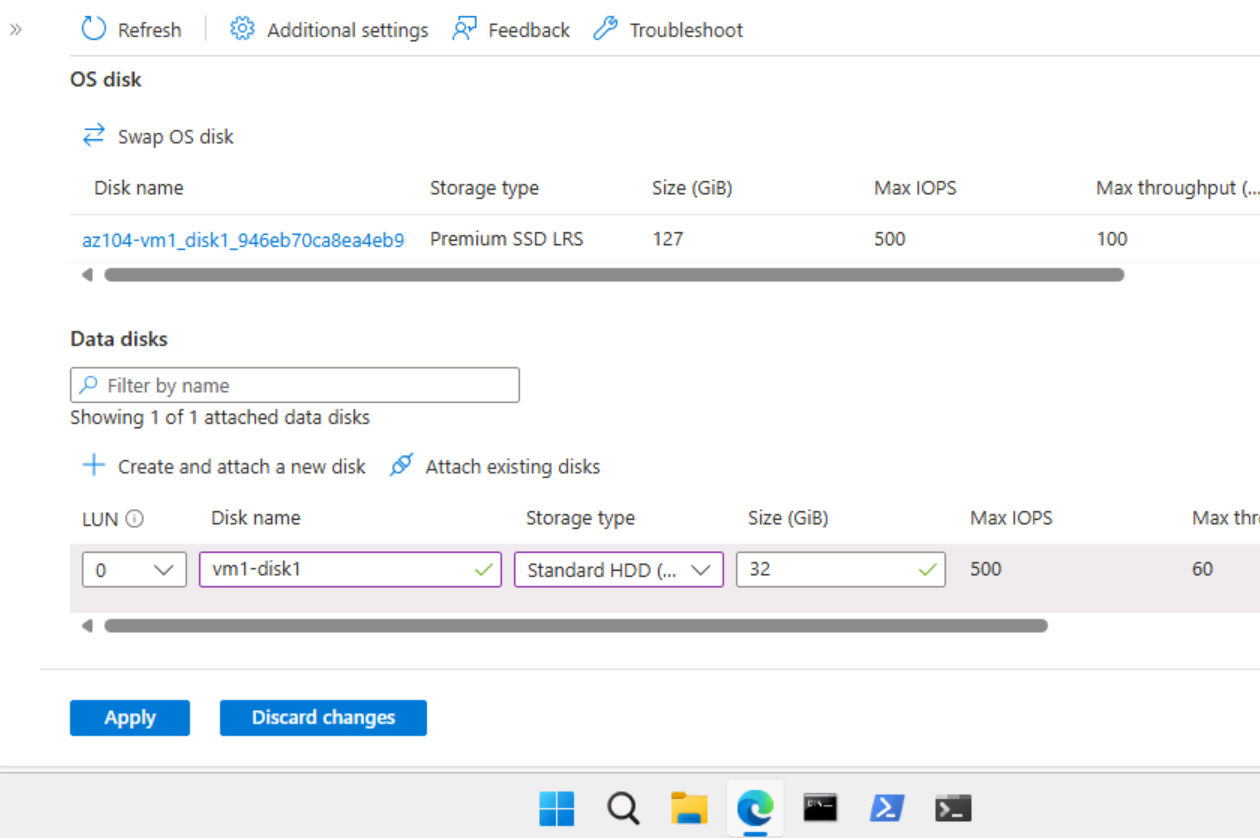
1. Navigated to az104-vm1 > Availability + scale > Size
2. Changed VM size to D2ds\_v4 and confirmed resize
3. Added a new data disk:

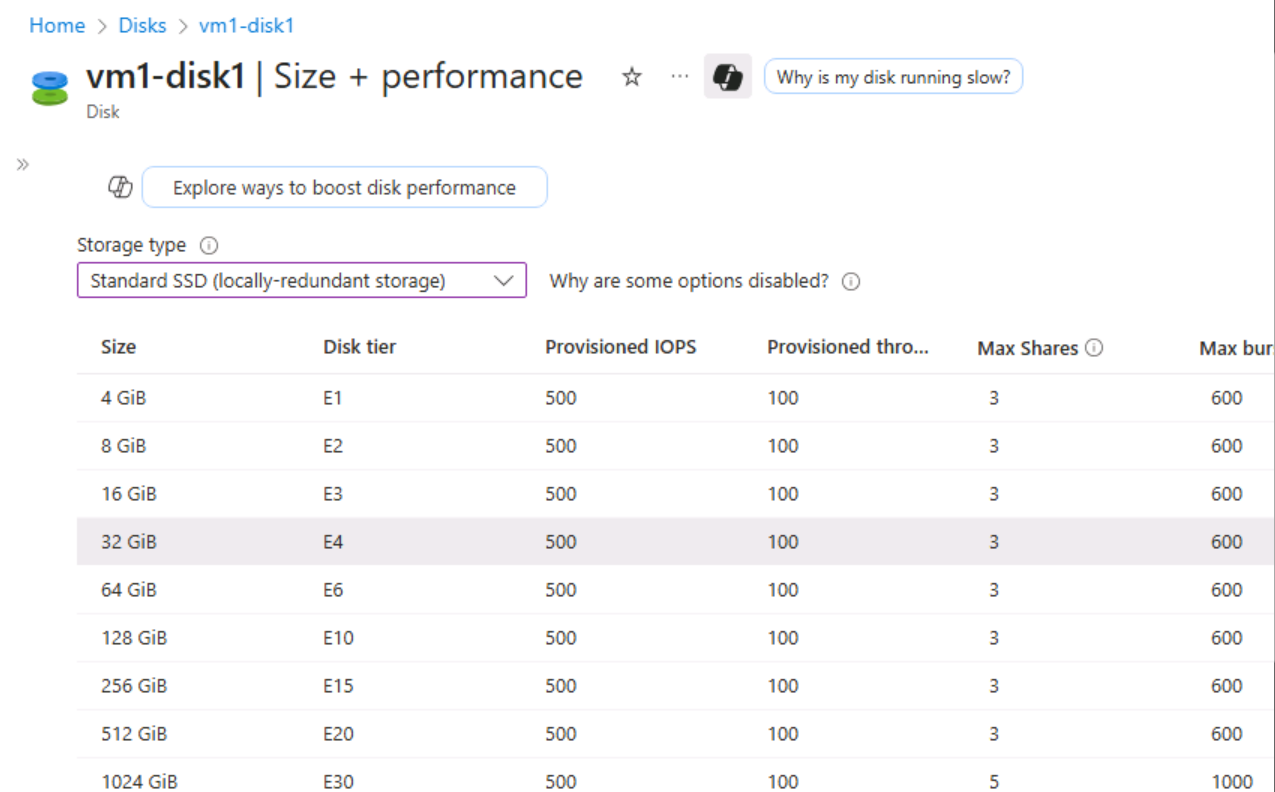
| **Setting** | **Value** |
| --- | --- |
| Disk Name | vm1-disk1 |
| Storage Type | Standard HDD |
| Size | 32 GiB |

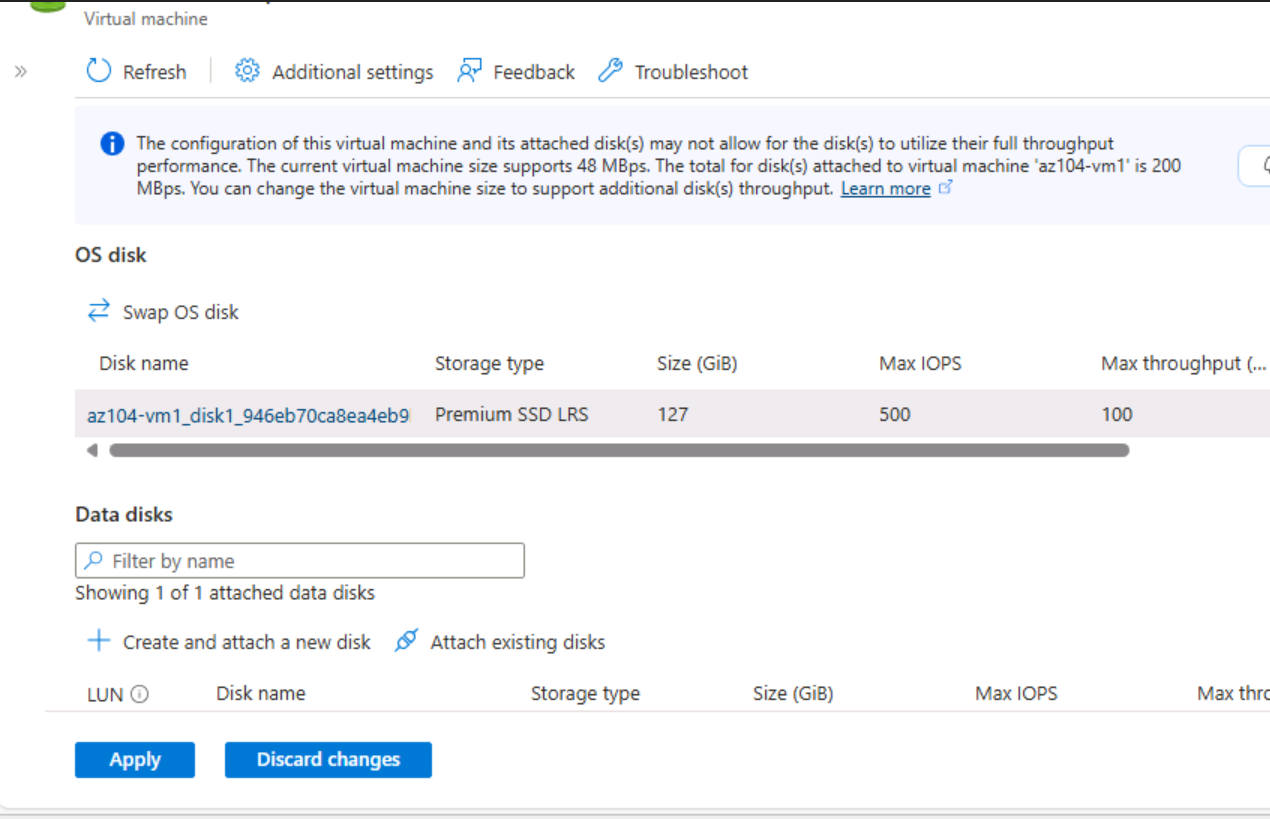
1. Detached the disk
2. Navigated to **Disks > vm1-disk1 > Size + performance**
3. Changed storage type to **Standard SSD**
4. Reattached disk to az104-vm1 and verified SSD upgrade

**Screenshot Checklist:**  
✅ VM size change confirmation (D2ds\_v4)

  
✅ Disk creation screen with Standard HDD

  
  
✅ Disk performance blade showing SSD upgrade

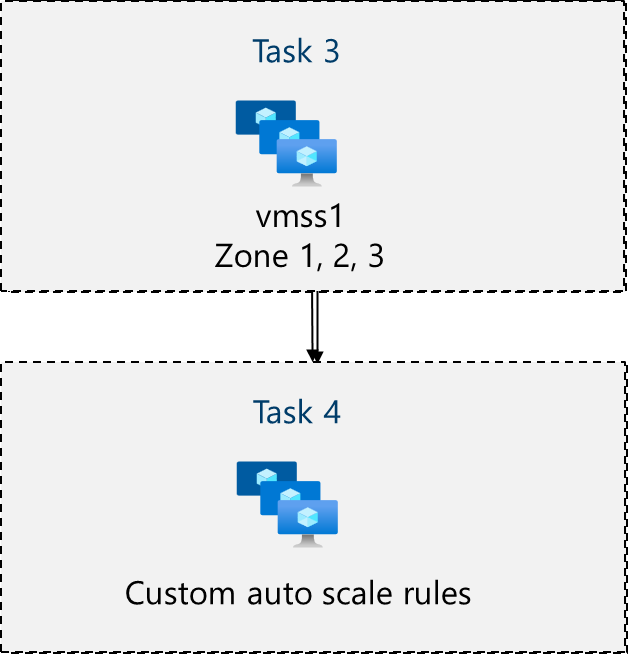
  
✅ Disk reattachment screen



**Notes:**

* Resizing VM is vertical scaling — useful for performance tuning
* Disk detachment preserves data for reuse
* SSD upgrade improves IOPS and latency

**🧩 Task 3: Create and Configure Azure Virtual Machine Scale Sets**



**Objective:**  
Deploy a VM Scale Set across multiple availability zones with networking, NSG, and load balancing.

**Steps Taken:**

1. Navigated to **Virtual Machine Scale Sets > + Create**
2. Configured the following on the Basics tab:

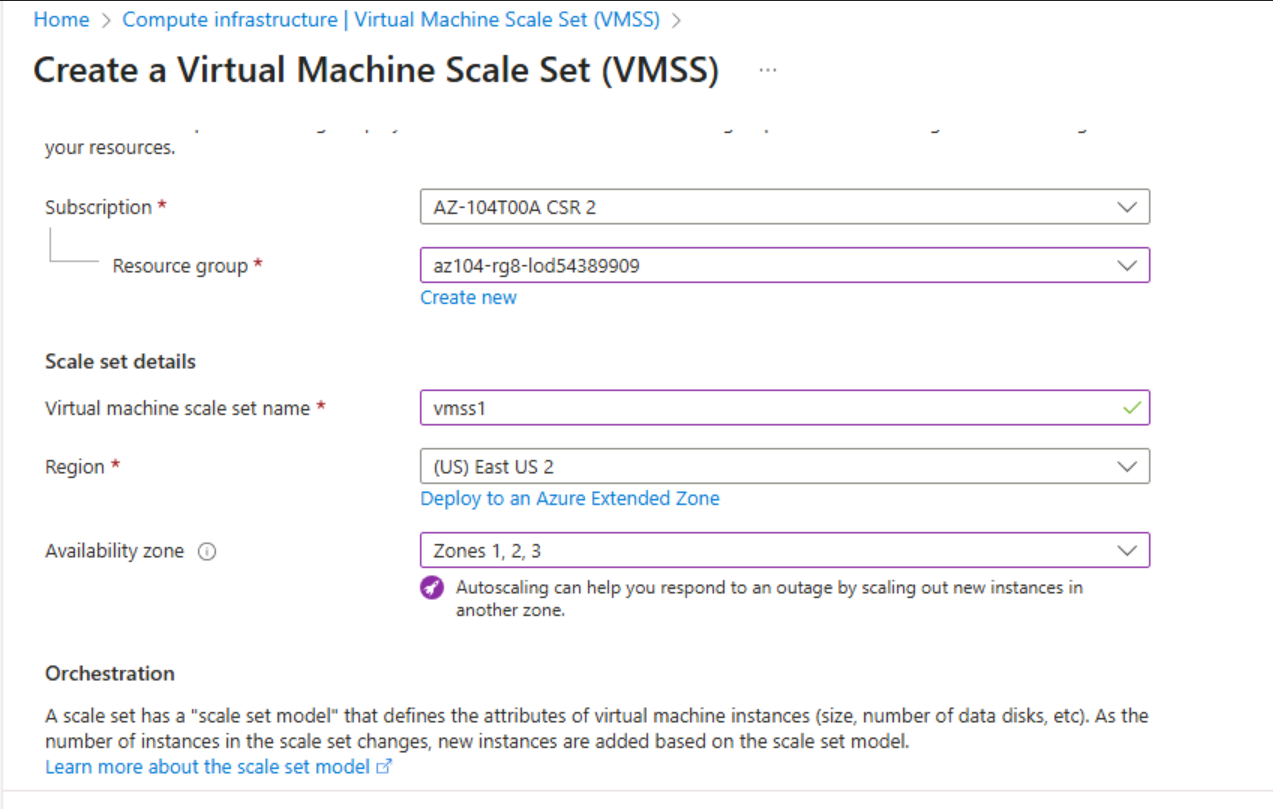
| **Setting** | **Value** |
| --- | --- |
| Resource Group | az104-rg8 |
| Scale Set Name | vmss1 |
| Region | East US |
| Availability Zones | 1, 2, 3 |
| Orchestration Mode | Uniform |
| Image | Windows Server 2019 Datacenter |
| Size | Standard D2s\_v3 |
| Username | localadmin |
| Password | [Secure] |

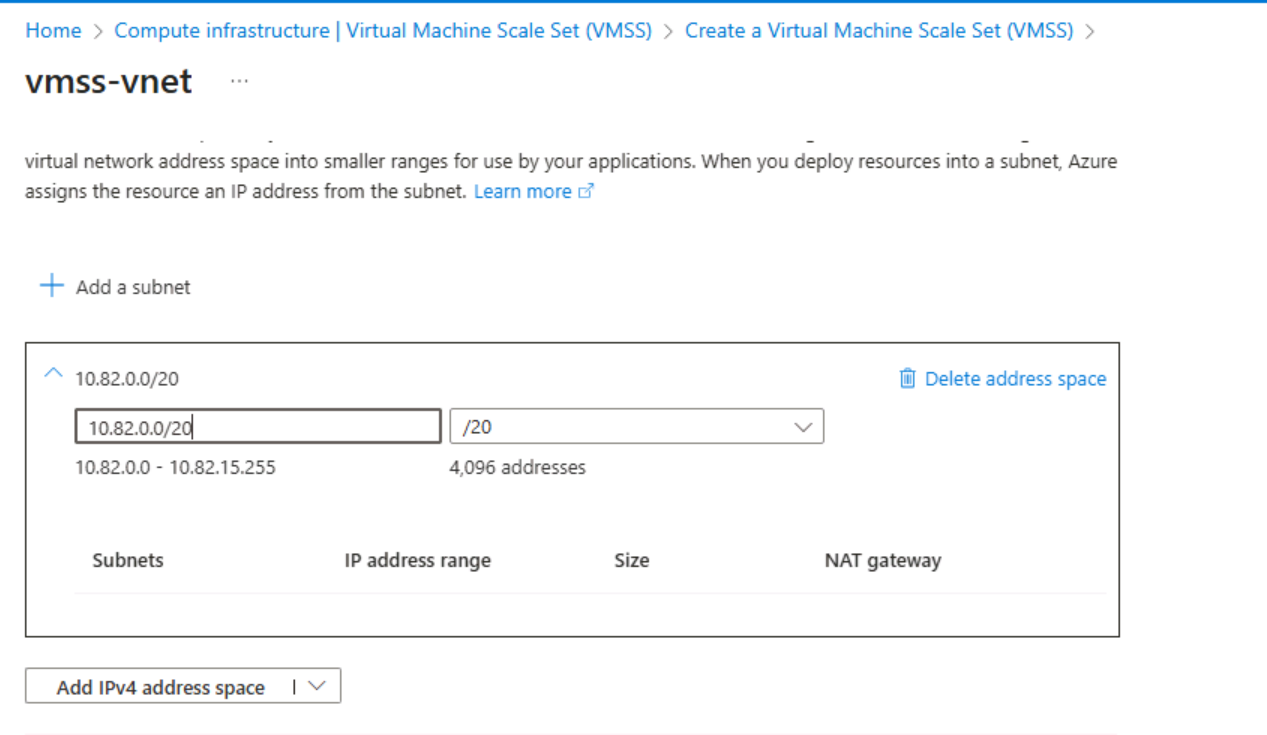
1. Accepted defaults on Spot and Disks tabs
2. Edited virtual network:

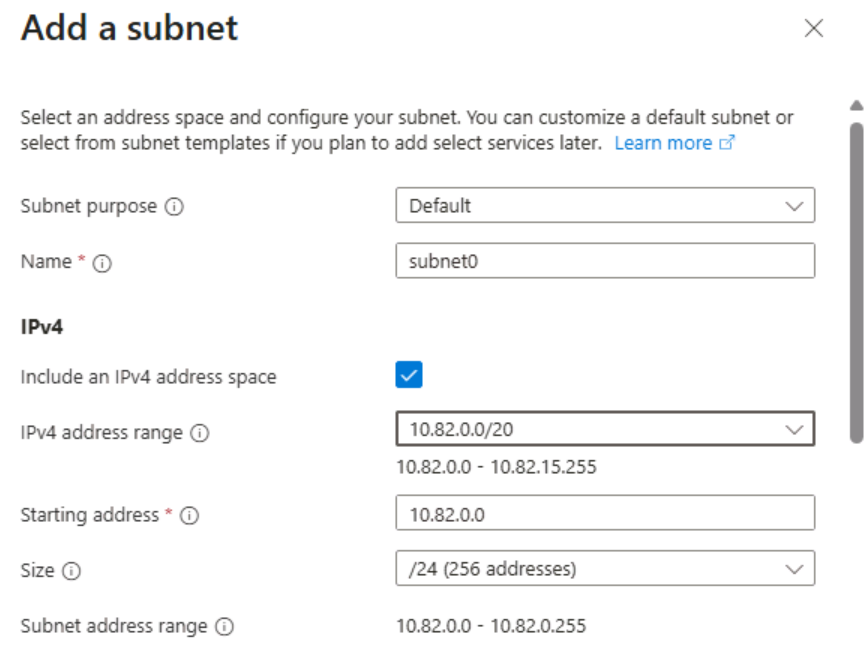
| **Setting** | **Value** |
| --- | --- |
| VNet Name | vmss-vnet |
| Address Range | 10.82.0.0/20 |
| Subnet Name | subnet0 |
| Subnet Range | 10.82.0.0/24 |

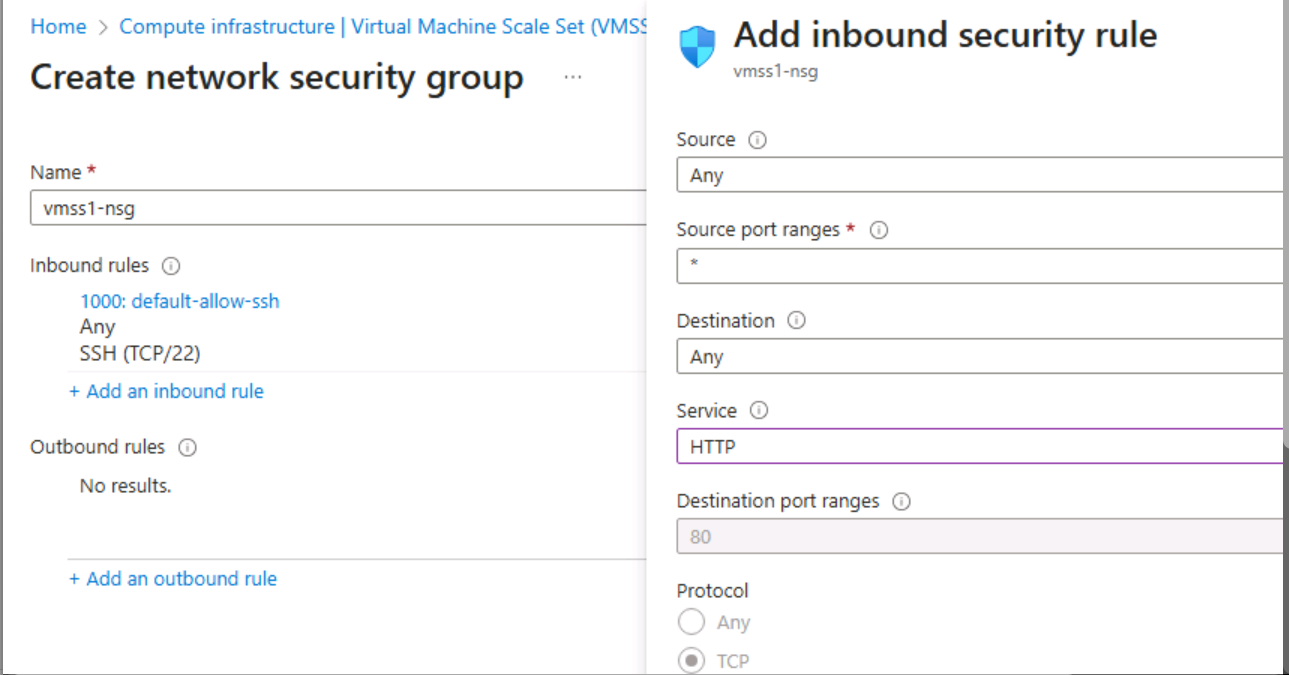
1. Created NSG vmss1-nsg with inbound HTTP rule
2. Enabled Public IP
3. Created Load Balancer vmss-lb
4. Disabled Boot Diagnostics
5. Validated and deployed

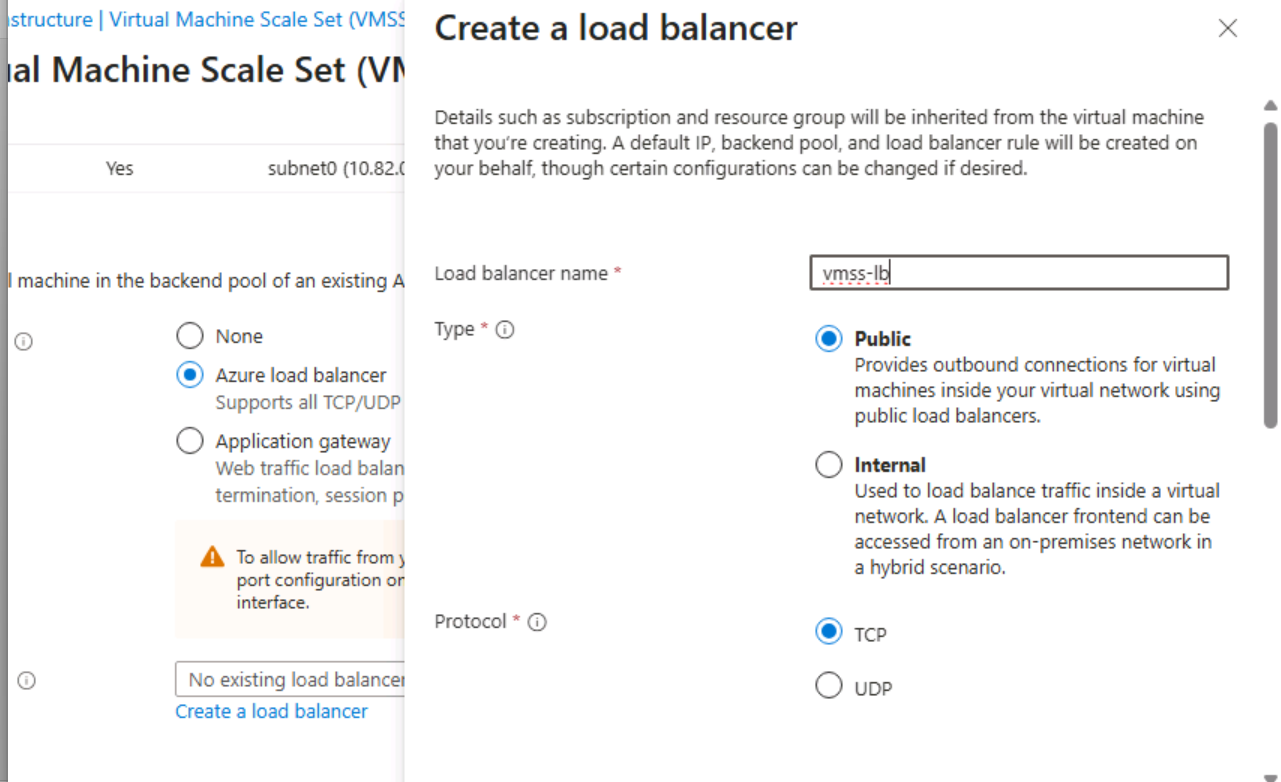
**Screenshot Checklist:**  
✅ VMSS creation page with zones selected

  
✅ VNet and subnet configuration



  
✅ NSG creation with HTTP rule

  
✅ Load balancer setup

  
✅ Final Review + Create screen

✅ Deployment complete confirmation  
✅ VMSS resource overview

**Notes:**

* VMSS simplifies horizontal scaling
* NSG and load balancer setup ensures secure, scalable access

**🧩 Task 4: Scale Azure Virtual Machine Scale Sets**

**Objective:**  
Configure autoscaling rules based on CPU metrics to scale VM instances dynamically.

**Steps Taken:**

1. Navigated to vmss1 > Availability + Scale > Scaling
2. Selected **Custom autoscale > Scale based on metric**
3. Created scale-out rule:

| **Setting** | **Value** |
| --- | --- |
| Metric | Percentage CPU |
| Operator | Greater than |
| Threshold | 70 |
| Duration | 10 min |
| Operation | Increase by 50% |
| Cooldown | 5 min |

1. Created scale-in rule:

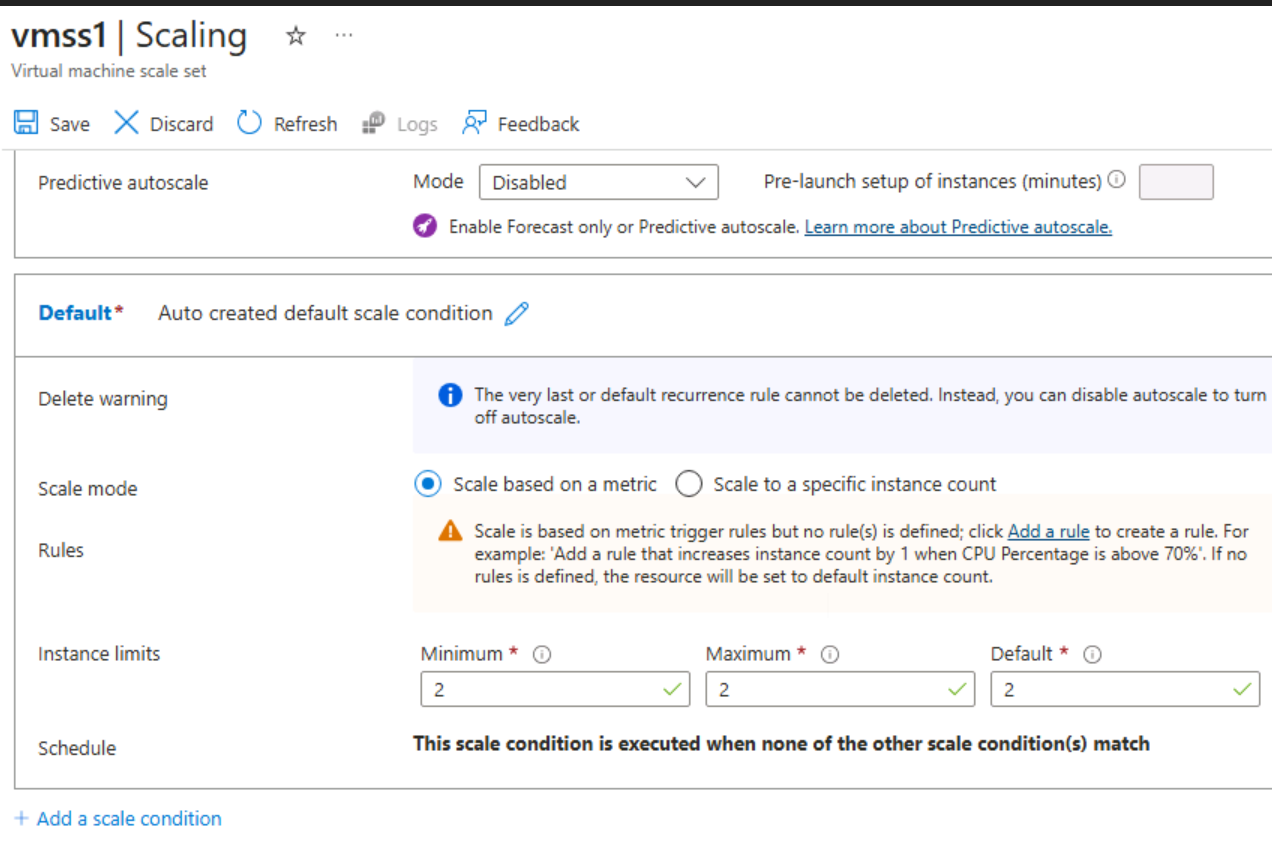
| **Setting** | **Value** |
| --- | --- |
| Operator | Less than |
| Threshold | 30 |
| Operation | Decrease by 50% |

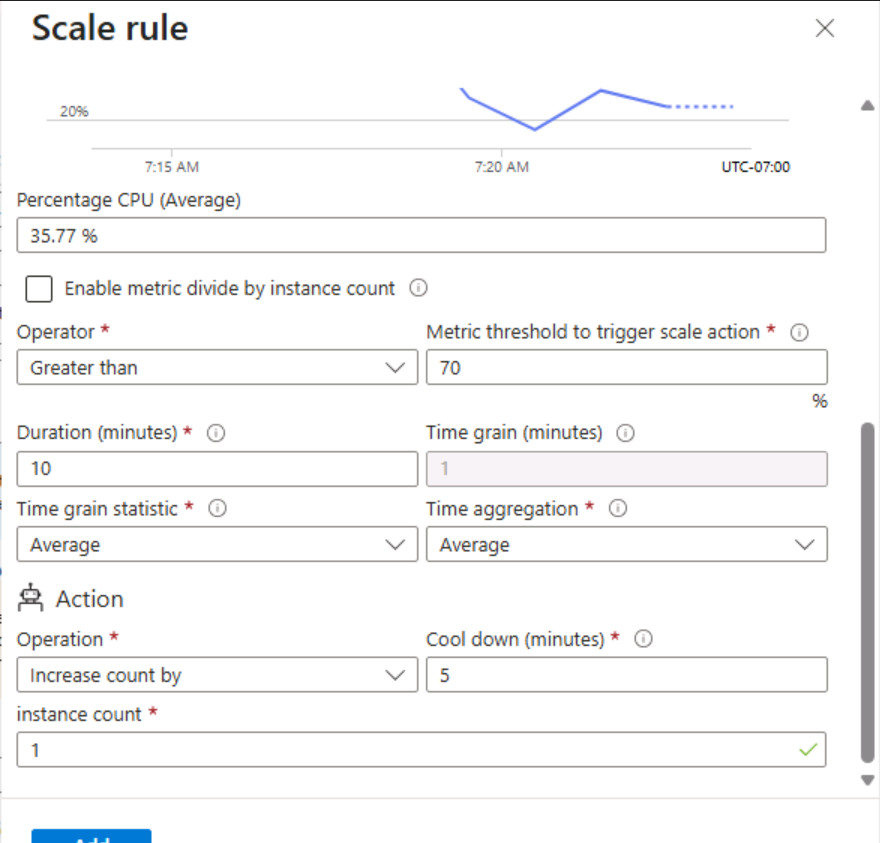
1. Set instance limits:

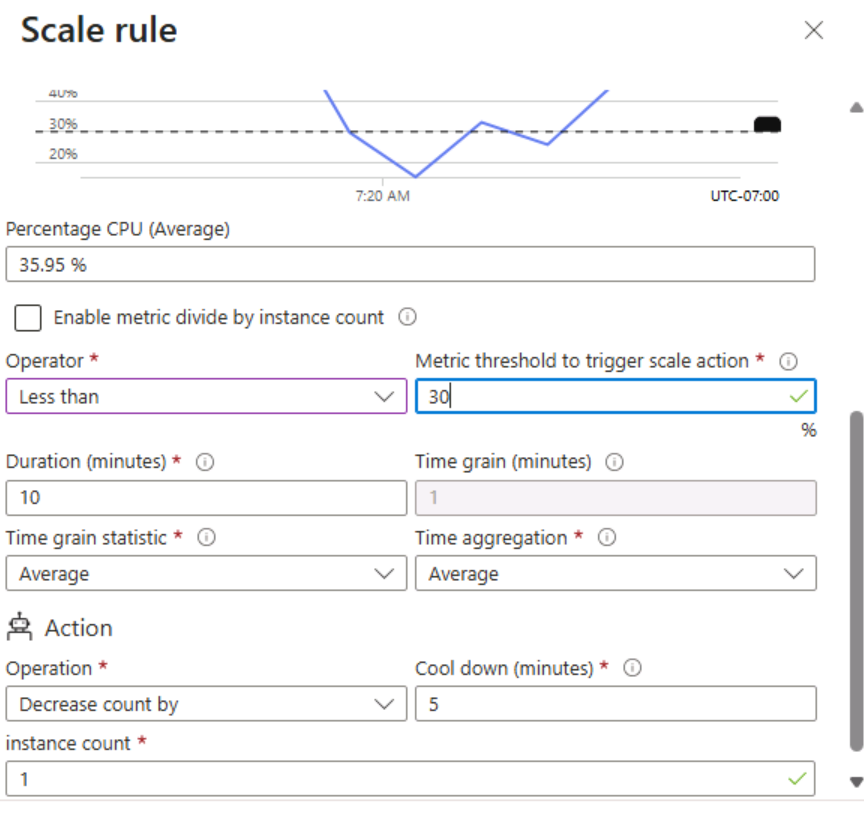
| **Setting** | **Value** |
| --- | --- |
| Minimum | 2 |
| Maximum | 10 |
| Default | 2 |

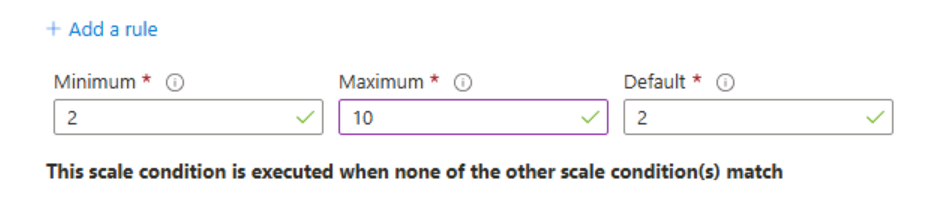
1. Saved all changes
2. Monitored instance count via **Instances** tab

**Screenshot Checklist:**  
✅ Scaling rules page with CPU metric

  
✅ Scale-out rule config

  
✅ Scale-in rule config

  
✅ Instance limits section



**Notes:**

* Autoscaling based on CPU ensures cost-efficiency and performance
* Instance limits prevent over-scaling or under-provisioning

**🧩 Task 5: Create a VM Using Azure PowerShell**

**Objective:**  
Deploy and manage a VM using PowerShell in Cloud Shell.

**Steps Taken:**

1. Launched Cloud Shell and selected PowerShell
2. Ran VM creation command:

New-AzVm `

-ResourceGroupName 'az104-rg8' `

-Name 'myPSVM' `

-Location 'East US' `

-Image 'Win2019Datacenter' `

-Zone '1' `

-Size 'Standard\_D2s\_v3' `

-Credential (Get-Credential)

1. Verified VM status:

Get-AzVM `

-ResourceGroupName 'az104-rg8' `

-Status

1. Deallocated VM:

Stop-AzVM `

-ResourceGroupName 'az104-rg8' `

-Name 'myPSVM'

1. Verified deallocation:

Get-AzVM `

-ResourceGroupName 'az104-rg8' `

-Status

**Screenshot Checklist:**  
✅ Cloud Shell with New-AzVM command  
✅ VM status showing “Running”  
✅ Stop-AzVM confirmation  
✅ VM status showing “Deallocated”

**Notes:**

* PowerShell offers full control over VM lifecycle
* Deallocation stops billing for compute resources

Let me know when you’re ready to drop Task 6 — I’ll slot it in and then help you wrap the doc with a clean summary and GitHub notes. We can also build a README template that explains your lab structure, screenshot policy, and reuse strategy. This is shaping up to be a killer portfolio piece.

🧩 Task 6: Create a Virtual Machine Using the Azure CLI

**Objective:**  
Deploy and manage a Linux VM using Bash in Azure Cloud Shell.

**Steps Taken:**

1. Launched Cloud Shell and selected **Bash**
2. Ran VM creation command:

az vm create \ --name myCLIVM \ --resource-group az104-rg8 \ --image Ubuntu2204 \ --admin-username localadmin \ --generate-ssh-keys

1. Verified VM status:

az vm show \ --name myCLIVM \ --resource-group az104-rg8 \ --show-details

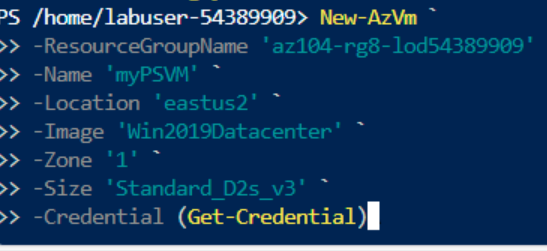
1. Deallocated VM:

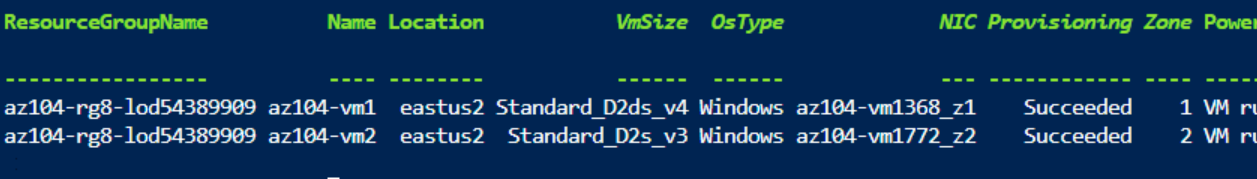
az vm deallocate \ --resource-group az104-rg8 \ --name myCLIVM

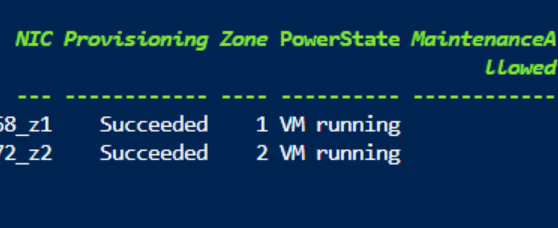
1. Confirmed deallocation:

az vm show \ --name myCLIVM \ --resource-group az104-rg8 \ --show-details

**Screenshot Checklist:**  
✅ Cloud Shell with az vm create command

  
✅ VM status showing VM Running





**Notes:**

* CLI offers fast, scriptable VM deployment
* Deallocation stops billing and releases non-static public IPs
* Ideal for automation and DevOps workflow