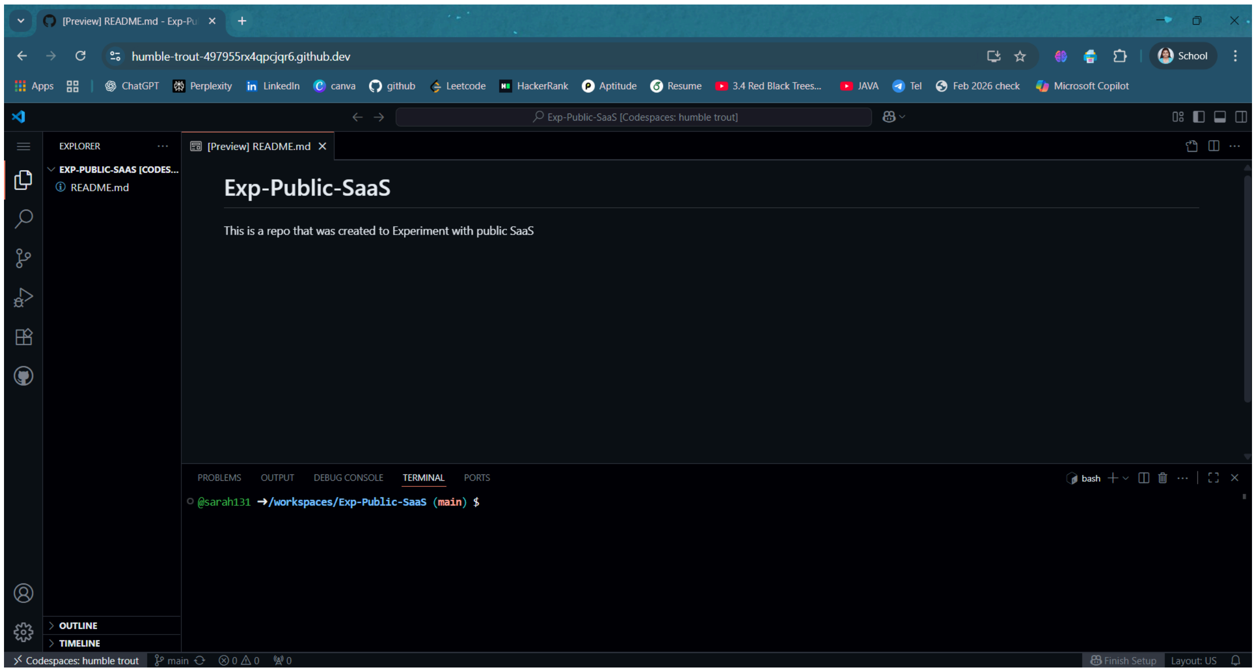
CLOUD COMPUTING LAB

Exp1: Experiment with public SaaS

Aim: Use GitHub (SaaS) and GitHub Codespaces (browser VS Code, SaaS) to create, edit, compile, run and version-control a C program without installing compilers/IDEs locally.

Step 1: Sign in to GitHub & create a repository

1. Go to github.com → sign in account
2. Click → New repository.
3. Choose Public (or Private, either is fine) → Create a repository. Using GitHub's web app (SaaS). This repo lives on GitHub's cloud.



Exp-Public-SaaS

This is a repo that was created to Experiment with public SaaS

```
@sarah131 →/workspaces/Exp-Public-SaaS (main) $
```

Step 2: Launch GitHub Codespaces (cloud dev environment)

1. On the new repo page, click the Code button.
2. Choose Open with Codespaces → New codespace. It spins up a Linux VM + VS Code in the browser. Just started a cloud-hosted IDE/terminal. No local installs needed.

Step 3: Create C program (in the browser)

1. In the Codespaces Explorer (left sidebar), click New File → name it main.c.
2. Paste this program:

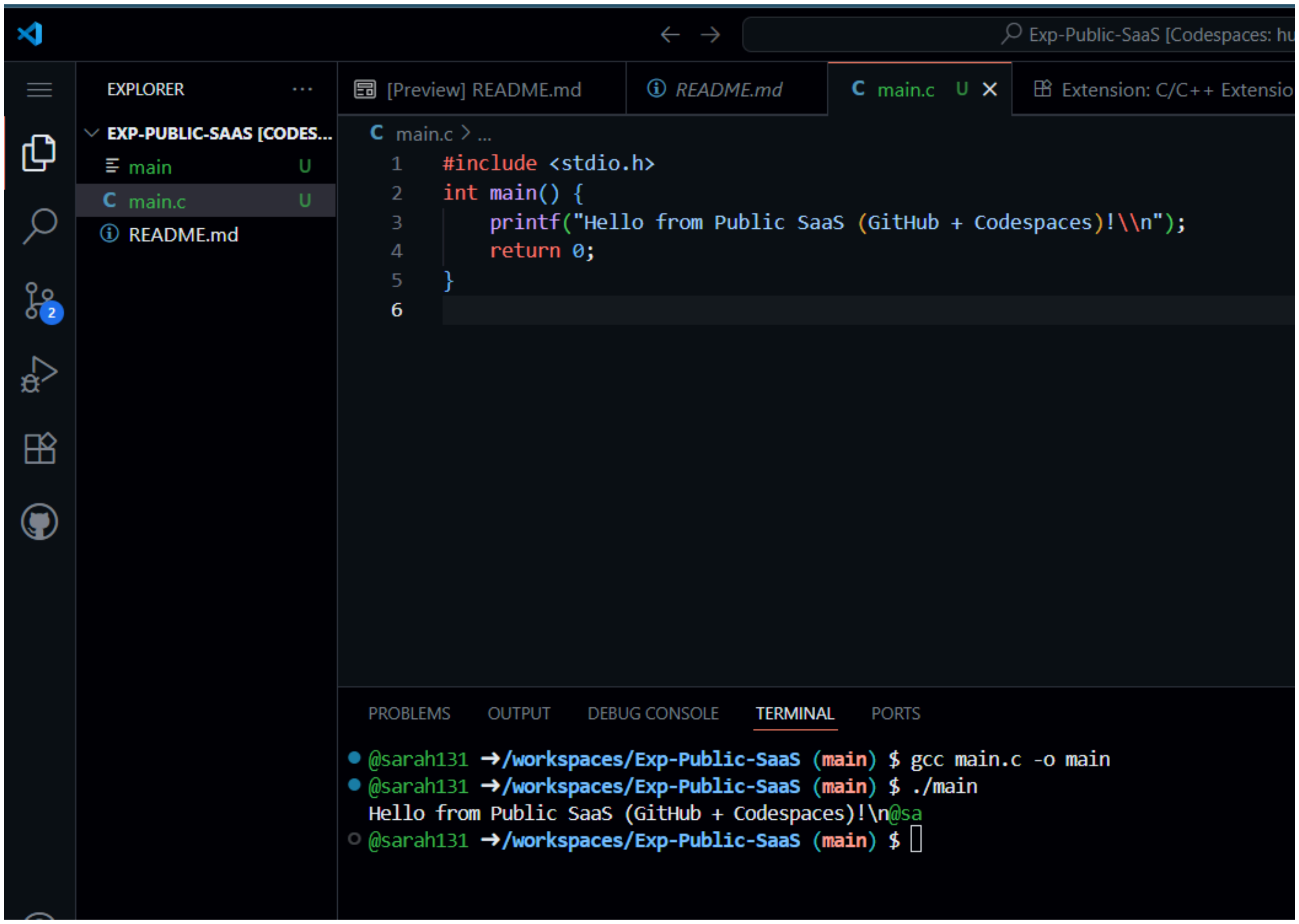
```
#include <stdio.h>
int main(){
printf("Hello from Public SaaS (GitHub + codespaces )!\\n");
```

```
 return 0;
}
```

3. Press **Ctrl+S** to save.

Step 4: Compile & run (inside Codespaces terminal)

1. Open the **Terminal** (View → Terminal).
2. Compile: `gcc main.c -o main`
3. Run: `./main`



```c
#include <stdio.h>
int main() {
    printf("Hello from Public SaaS (GitHub + Codespaces)!\\n");
    return 0;
}
```

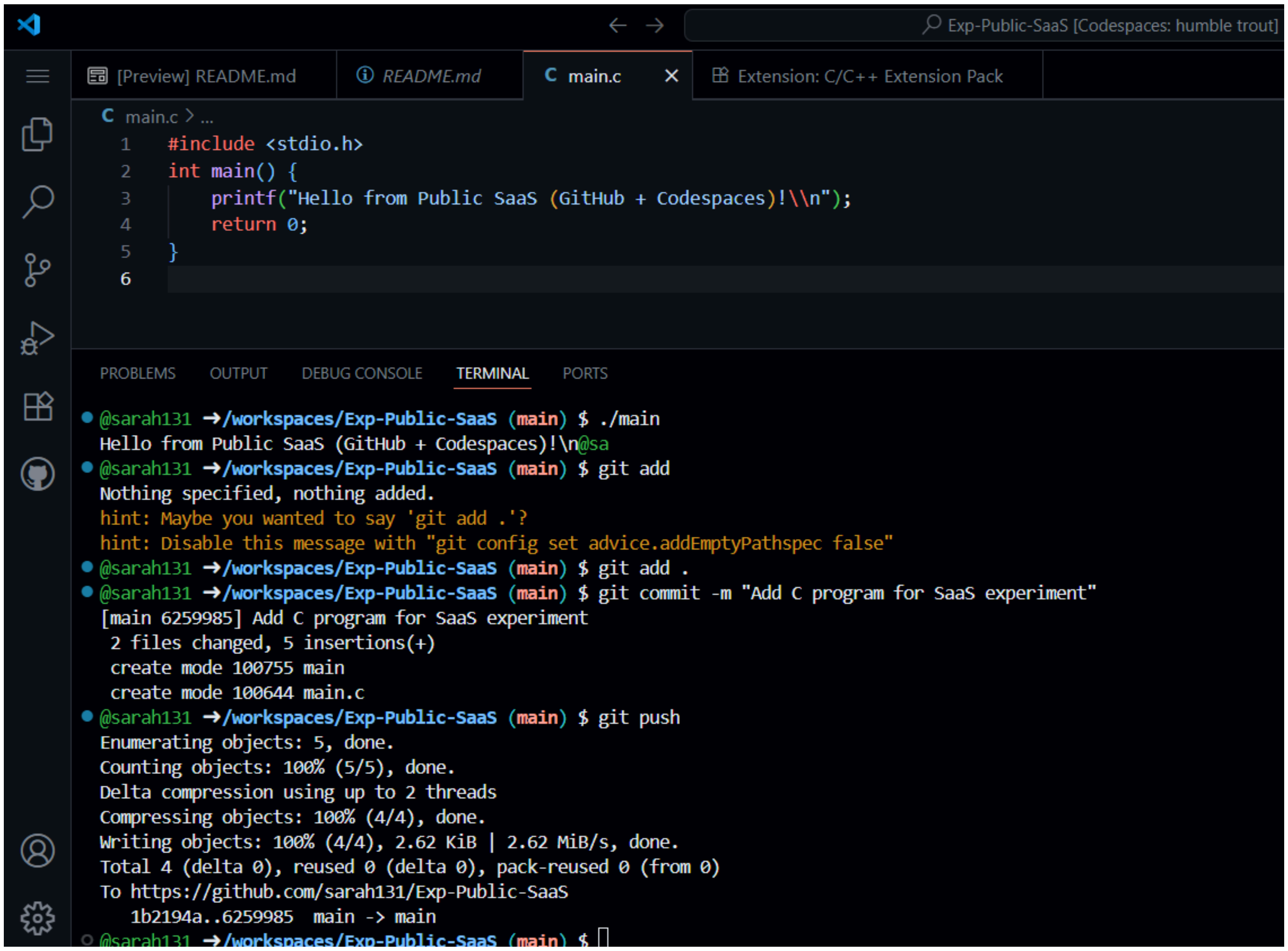
```
@sarah131 →/workspaces/Exp-Public-SaaS (main) $ gcc main.c -o main
@sarah131 →/workspaces/Exp-Public-SaaS (main) $ ./main
Hello from Public SaaS (GitHub + Codespaces)!\n@sa
@sarah131 →/workspaces/Exp-Public-SaaS (main) $
```

The compiler (gcc) and CPU/RAM are on GitHub's cloud VM, not your machine.

Step 5: Commit & push to GitHub (version control in the cloud)

In the terminal: `git add .`

```
git commit -m "Add C program for SaaS experiment"

git push
```



```c
#include <stdio.h>
int main() {
    printf("Hello from Public SaaS (GitHub + Codespaces)!\\n");
    return 0;
}

```

```
@sarah131 →/workspaces/Exp-Public-SaaS (main) $ ./main
Hello from Public SaaS (GitHub + Codespaces)!\n@sa
@sarah131 →/workspaces/Exp-Public-SaaS (main) $ git add
Nothing specified, nothing added.
hint: Maybe you wanted to say 'git add .'?
hint: Disable this message with "git config set advice.addEmptyPathspec false"
@sarah131 →/workspaces/Exp-Public-SaaS (main) $ git add .
@sarah131 →/workspaces/Exp-Public-SaaS (main) $ git commit -m "Add C program for SaaS experiment"
[main 6259985] Add C program for SaaS experiment
 2 files changed, 5 insertions(+)
 create mode 100755 main
 create mode 100644 main.c
@sarah131 →/workspaces/Exp-Public-SaaS (main) $ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 2.62 KiB | 2.62 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/sarah131/Exp-Public-SaaS
   1b2194a..6259985  main -> main
@sarah131 →/workspaces/Exp-Public-SaaS (main) $
```

Later, return to your repo → **Code** → **Open with Codespaces** → pick your existing codespace.

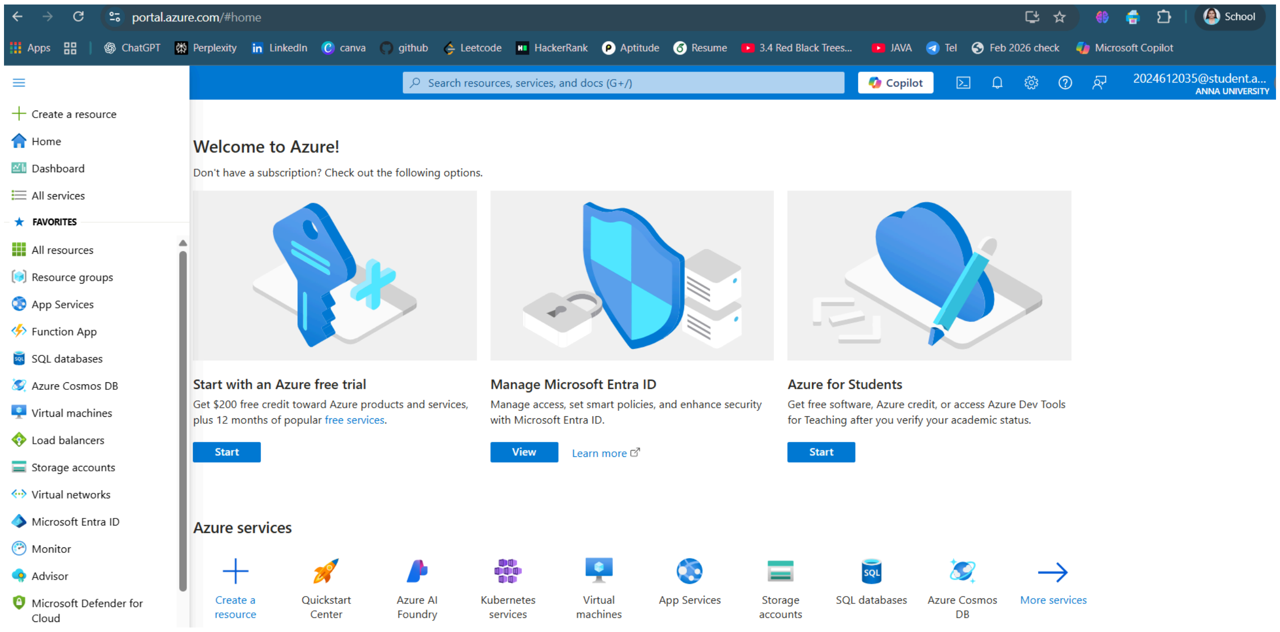
Exp2: Create a software using public PaaS

Aim: To use Microsoft Azure and creating a webpage that shows Public PaaS

Login to this website portal.azure.com

Step 1: Install tools

1. Install VS Code.
2. Install Node.js LTS. (Check: node -v, npm -v)
3. Install Azure CLI. (Check: az --version)
4. In VS Code, go to Extensions → install Azure Tools (this includes *Azure App Service*, *Azure Account*, etc.).



Welcome to Azure!

Don't have a subscription? Check out the following options.

Start with an Azure free trial

Get $200 free credit toward Azure products and services, plus 12 months of popular free services.

Manage Microsoft Entra ID

Manage access, set smart policies, and enhance security with Microsoft Entra ID.

Azure for Students

Get free software, Azure credit, or access Azure Dev Tools for Teaching after you verify your academic status.

Azure services

Step 2: Create a simple Node.js app in VS Code

Open a new folder in VS Code → Terminal → run:

```
 npm init -y
npm install express
```

Create app.js:

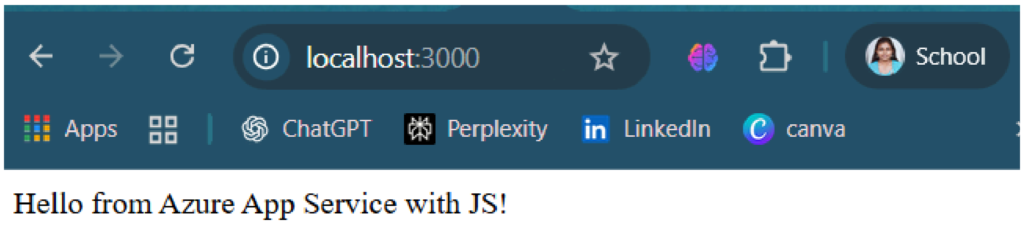
```
 const express = require("express")
const app = express()
const port = process.env.PORT || 3000
app.get("/", (req, res) => res.send("Hello from Azure App Service via VS Code!"))
app.listen(port, () => console.log("Listening on", port))
```

In package.json, add:

```
 {
 "name": "azure-demo",
 "version": "1.0.0",
 "main": "app.js",
 "scripts": {
   "start": "node app.js"
 },
 "dependencies": {
   "express": "^4.18.2"
 }
}
```

Test locally: node app.js

Visit http://localhost:3000.



Hello from Azure App Service with JS!

Exp3: Experiment storage services in cloud

Step 1: Create a Storage Account

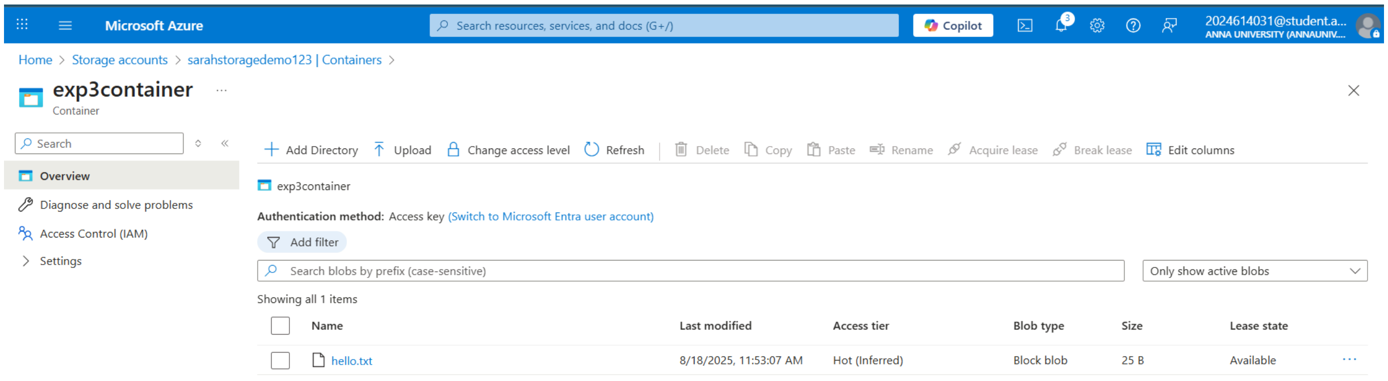
1. Login to Azure Portal, Search Storage accounts → click + Create.
2. Enter details:
   - Resource Group: rg-storage-demo
   - Storage Account Name: mystorageexp3 (must be unique)
   - Region: Central India (or nearest)
   - Performance: Standard
   - Redundancy: Locally-redundant storage (LRS)
3. Click Review + Create → Create.

Step 2: Create a Container

1. Open your storage account → click Containers.
2. Click + Container.
   - Name: exp3container
   - Public Access: Private (no anonymous access)
3. Click Create.

Step 3: Upload a File

1. Inside the container, click Upload.
2. Select a file from your computer (e.g., hello.txt).
3. The file is now stored in Azure Blob Storage (but private).


Home > Storage accounts > sarahstoragedemo123 | Containers >

exp3container
Container

Overview
Diagnose and solve problems
Access Control (IAM)
Settings

Add Directory | Upload | Change access level | Refresh | Delete | Copy | Paste | Rename | Acquire lease | Break lease | Edit columns

exp3container

Authentication method: Access key (Switch to Microsoft Entra user account)

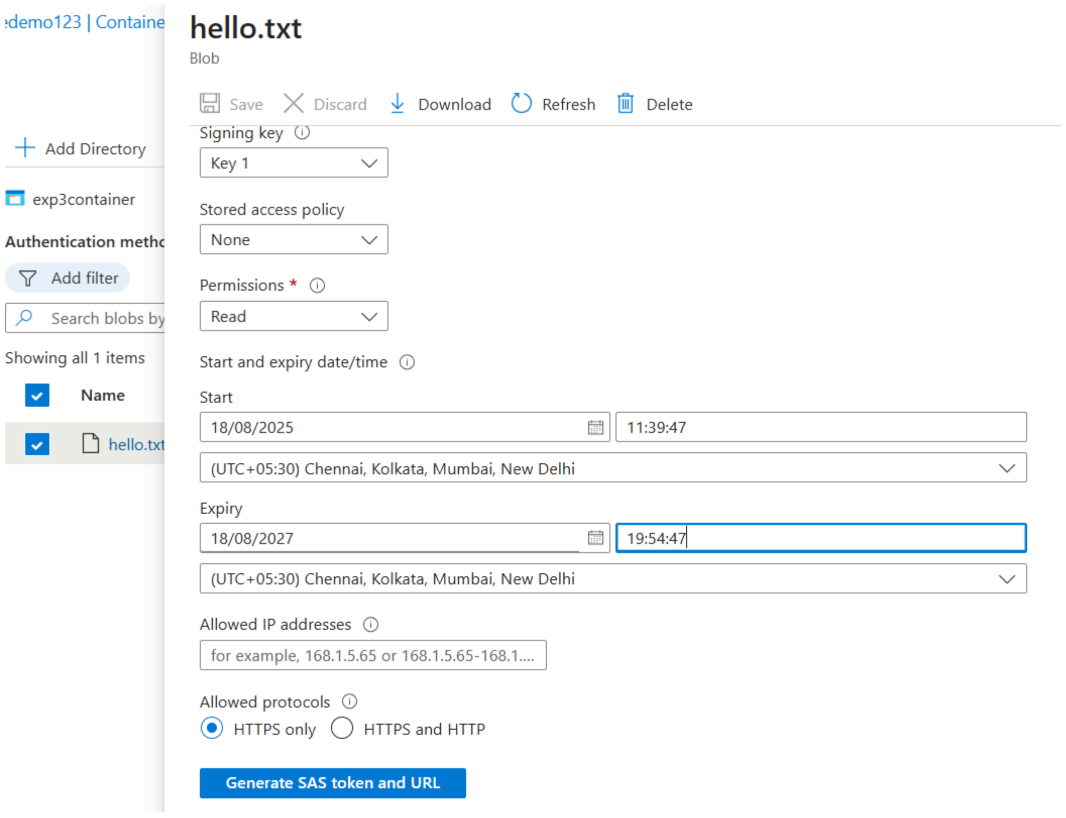
Only show active blobs

Showing all 1 items

Name	Last modified	Access tier	Blob type	Size	Lease state
hello.txt	8/18/2025, 11:53:07 AM	Hot (Inferred)	Block blob	25 B	Available

Step 4: Generate a SAS Link

1. Click on the uploaded file → at the top, click **Generate SAS**.
2. In the dialog:
    - Permissions: Read
    - Expiry Time: Set to a future date
    - Click **Generate SAS and URL**.



Signing key: Key 1

Stored access policy: None

Permissions *: Read

Start and expiry date/time

Start: 18/08/2025 11:39:47 (UTC+05:30) Chennai, Kolkata, Mumbai, New Delhi

Expiry: 18/08/2027 19:54:47 (UTC+05:30) Chennai, Kolkata, Mumbai, New Delhi

Allowed IP addresses: for example, 168.1.5.65 or 168.1.5.65-168.1....

Allowed protocols: [✓ HTTPS only] HTTPS and HTTP

3. Copy the **Blob SAS URL** provided.
   Blob SAS Token:

sp=r&st=2025-08-18T06:09:47Z&se=2027-08-18T14:24:47Z&spr=https&sv=2024-11-04&sr=b&sig=VsLizSo4Wr3Tl0ZLhfBPmKaWc1oX4XXfmdRcGK7Hyjc%3D

Blob SAS URL:

https://sarahstoragedemo123.blob.core.windows.net/exp3container/hello.txt?sp=r&st=2025-08-18T06:09:47Z&se=2027-08-18T14:24:47Z&spr=https&sv=2024-11-04&sr=b&sig=VsLizSo4Wr3Tl0ZLhfBPmKaWc1oX4XXfmdRcGK7Hyjc%3D

Step 5: Access File Securely

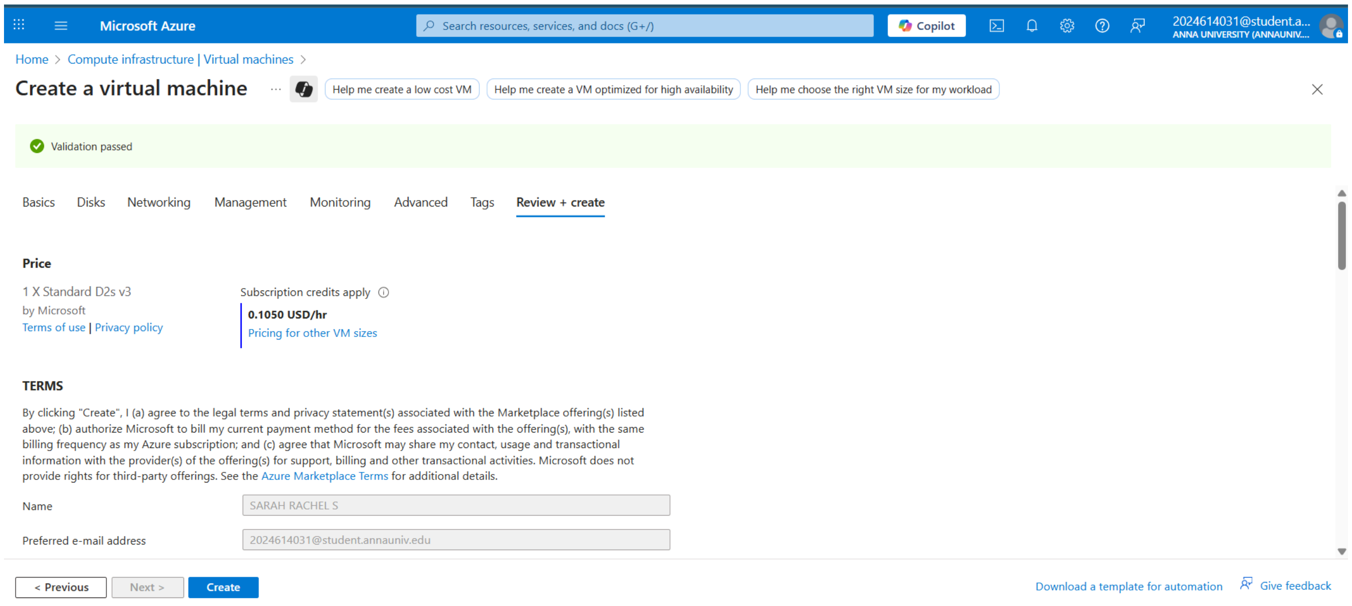
1. Paste the SAS URL in your browser.
2. The file opens/downloads successfully even though the container is private.

Exp 4: Create VMs in Public cloud platforms

Step 1: Login to Azure Portal - Open your browser and go to https://portal.azure.com

Click on **Virtual Machines** from the list.
Step 2: Create a New Virtual Machine
Click + Create → Azure virtual machine.
Configure Basics:

- Subscription: Select your free subscription.
- Resource Group: Either create a new one (ex: exp4-rg) or use existing.
- Virtual machine name: Example → exp4-vm.
- Region: Select the region
- Image (OS): Ubuntu Server 20.04 LTS (Linux)
- Size: Choose a B1s (free tier eligible) to save credits.
- Networking Setup: give default
- Authentication type:
  - Select Password (easy for learning).
  - Give a username (ex: azureuser) and a strong password.
- Ensure Public inbound ports → Allow selected ports.
- Choose SSH (22) for Linux
- Review + Create
- Skip Disk/Management tabs (default is fine).
- Click **Review + Create**.
- Wait for validation → then click **Create**.


Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
2024614031@student.a...
ANNA UNIVERSITY (ANNAUNIV...
Home > Compute infrastructure | Virtual machines >
Help me create a low cost VM
Help me create a VM optimized for high availability
Help me choose the right VM size for my workload
Basics
Disks
Networking
Management
Monitoring
Advanced
Tags
Review + create
< Previous
Next >
Create
Download a template for automation
Give feedback

Create a virtual machine

Validation passed

Price

1 X Standard D2s v3
by Microsoft
Terms of use | Privacy policy

Subscription credits apply
0.1050 USD/hr
Pricing for other VM sizes

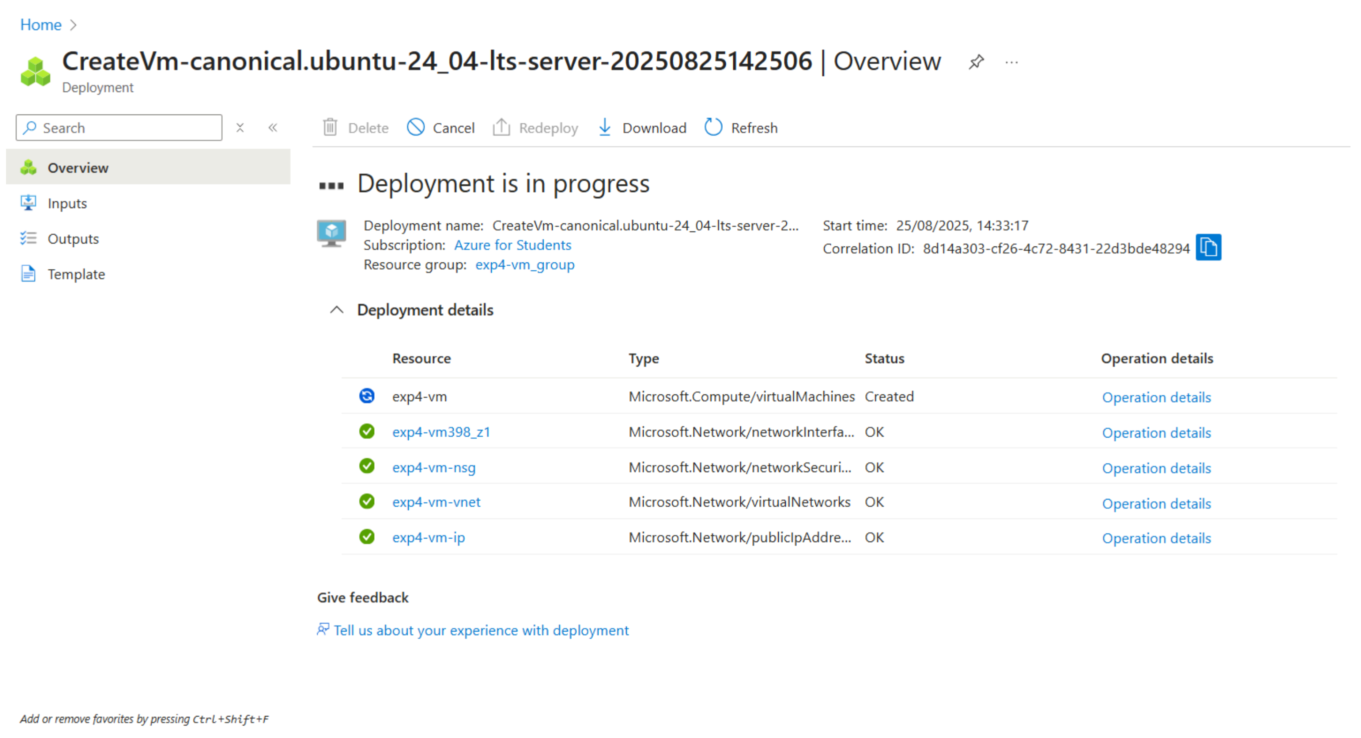
TERMS

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. See the Azure Marketplace Terms for additional details.

Name	SARAH RACHEL S
Preferred e-mail address	2024614031@student.annauniv.edu

Step 3: Deployment & Connect

While Azure deploys VM, go to your VM → **Overview page**. Copy the **Public IP address**.


Home >

CreateVm-canonical.ubuntu-24_04-lts-server-20250825142506 | Overview

Deployment

Deployment is in progress

Deployment name: CreateVm-canonical.ubuntu-24_04-lts-server-2...
Subscription: Azure for Students
Resource group: exp4-vm_group

Start time: 25/08/2025, 14:33:17
Correlation ID: 8d14a303-cf26-4c72-8431-22d3bde48294

Deployment details

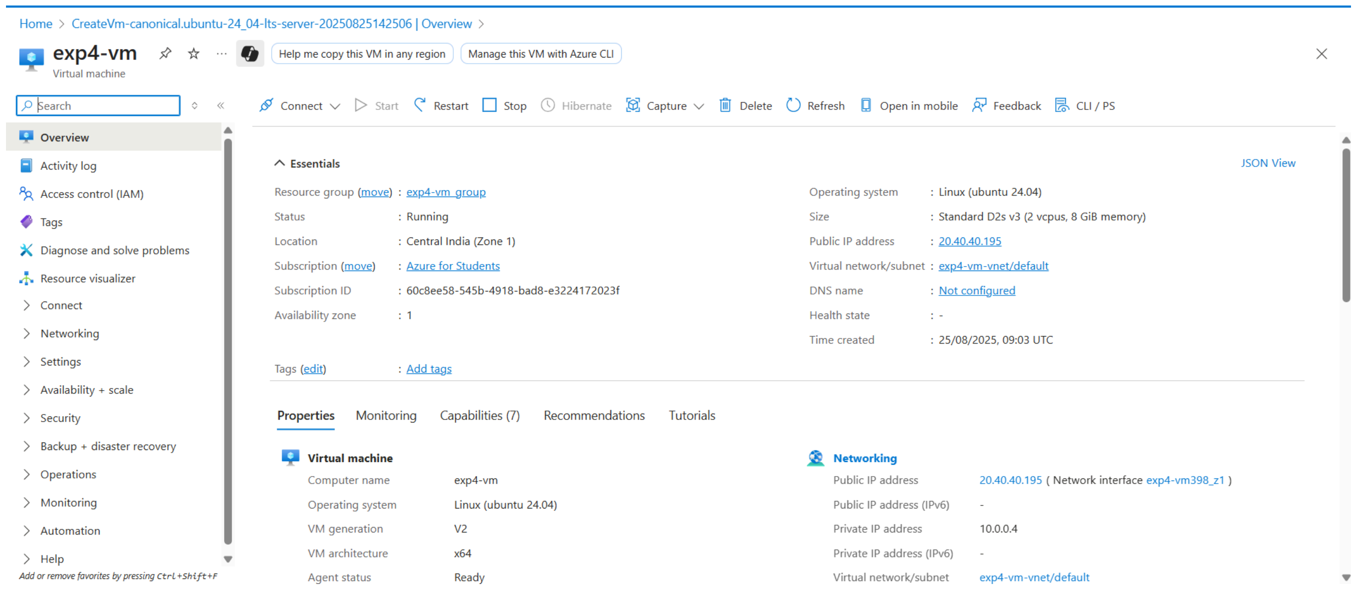
Resource	Type	Status	Operation details
exp4-vm	Microsoft.Compute/virtualMachines	Created	Operation details
exp4-vm398_z1	Microsoft.Network/networkInterfa...	OK	Operation details
exp4-vm-nsg	Microsoft.Network/networkSecuri...	OK	Operation details
exp4-vm-vnet	Microsoft.Network/virtualNetworks	OK	Operation details
exp4-vm-ip	Microsoft.Network/publicIpAddre...	OK	Operation details

Give feedback

Tell us about your experience with deployment

Add or remove favorites by pressing Ctrl+Shift+F

Click - Go to resources:


Home > CreateVm-canonical.ubuntu-24_04-lts-server-20250825142506 | Overview >

exp4-vm

Virtual machine

Add or remove favorites by pressing Ctrl+Shift+F

Essentials

Resource group (move)	: exp4-vm_group	Operating system	: Linux (ubuntu 24.04)
Status	: Running	Size	: Standard D2s v3 (2 vcpus, 8 GiB memory)
Location	: Central India (Zone 1)	Public IP address	: 20.40.40.195
Subscription (move)	: Azure for Students	Virtual network/subnet	: exp4-vm-vnet/default
Subscription ID	: 60c8ee58-545b-4918-bad8-e3224172023f	DNS name	: Not configured
Availability zone	: 1	Health state	: -
		Time created	: 25/08/2025, 09:03 UTC
Tags (edit)	: Add tags		

Virtual machine

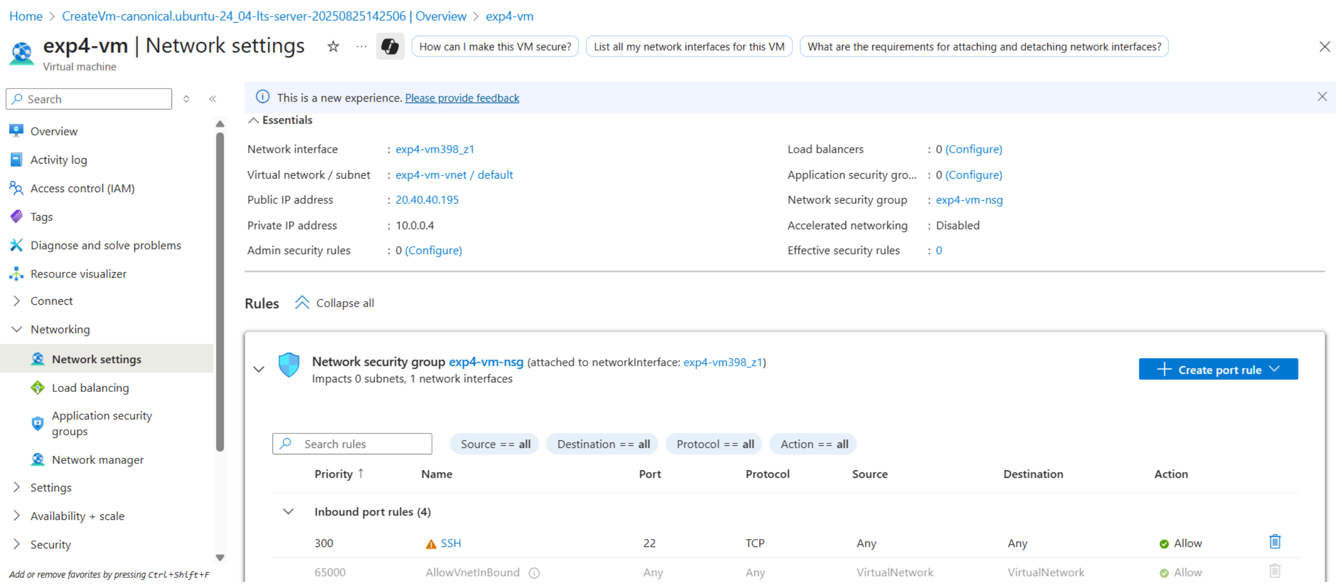
Computer name	exp4-vm
Operating system	Linux (ubuntu 24.04)
VM generation	V2
VM architecture	x64
Agent status	Ready

Networking

Public IP address	20.40.40.195 (Network interface exp4-vm398_z1)
Public IP address (IPv6)	-
Private IP address	10.0.0.4
Private IP address (IPv6)	-
Virtual network/subnet	exp4-vm-vnet/default

Step 4: Verify VM is Running

Go to networking > Network Setting > Check port rules for SSH



Home > CreateVm-canonical.ubuntu-24_04-lts-server-20250825142506 | Overview > exp4-vm

exp4-vm | Network settings
Virtual machine

This is a new experience. Please provide feedback

Essentials

Network interface	: exp4-vm398_z1	Load balancers	: 0 (Configure)
Virtual network / subnet	: exp4-vm-vnet / default	Application security gro...	: 0 (Configure)
Public IP address	: 20.40.40.195	Network security group	: exp4-vm-nsg
Private IP address	: 10.0.0.4	Accelerated networking	: Disabled
Admin security rules	: 0 (Configure)	Effective security rules	: 0

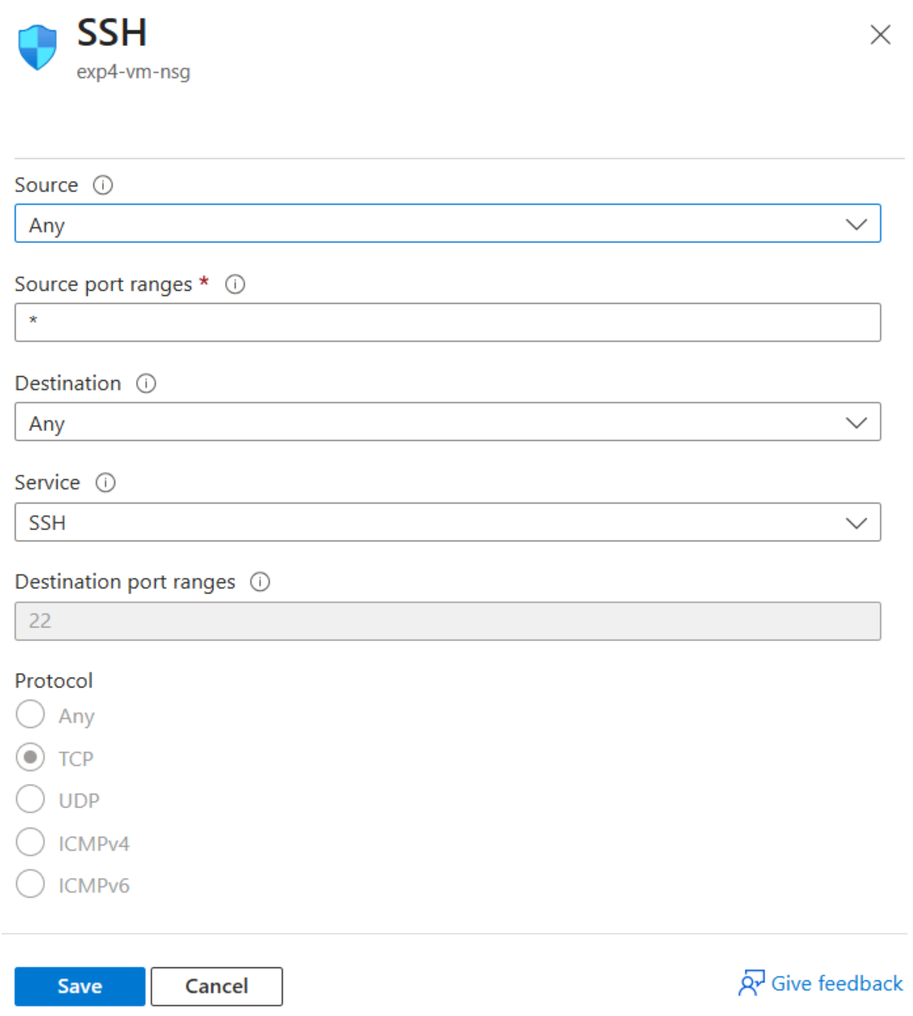
Rules

Network security group exp4-vm-nsg (attached to networkInterface: exp4-vm398_z1)
Impacts 0 subnets, 1 network interfaces

Source == all Destination == all Protocol == all Action == all

Inbound port rules (4)

Priority	Name	Port	Protocol	Source	Destination	Action
300	SSH	22	TCP	Any	Any	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow


SSH

exp4-vm-nsg

Source: Any

Source port ranges *: *

Destination: Any

Service: SSH

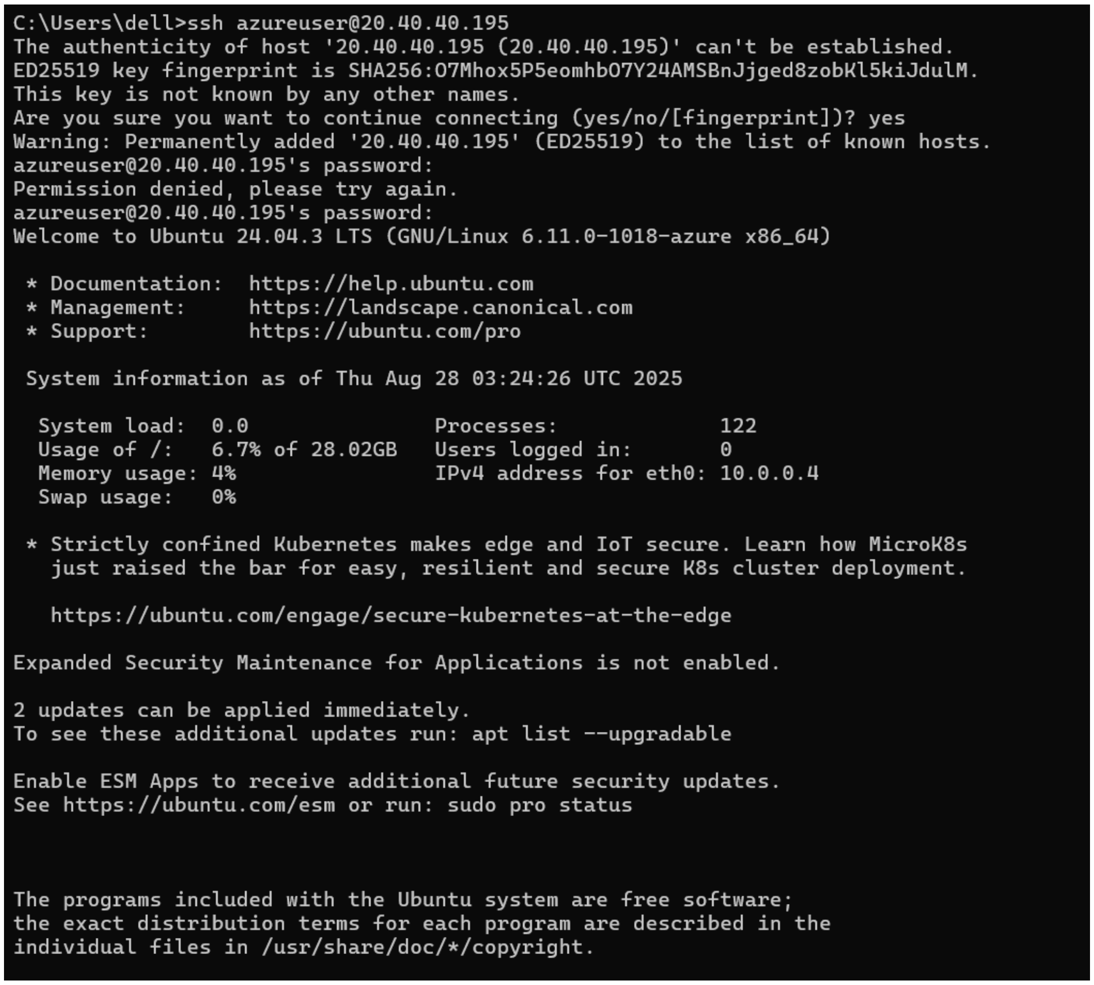
Destination port ranges: 22

Protocol
- ( ) Any
- (●) TCP
- ( ) UDP
- ( ) ICMPv4
- ( ) ICMPv6

If Linux VM → Open **VS Code terminal** or Command Prompt and run:

```
ssh azureuser@<Public-IP> => ssh azureuser@20.40.40.195
```

For Linux: run `ls`, `pwd`, `uname -a` etc.

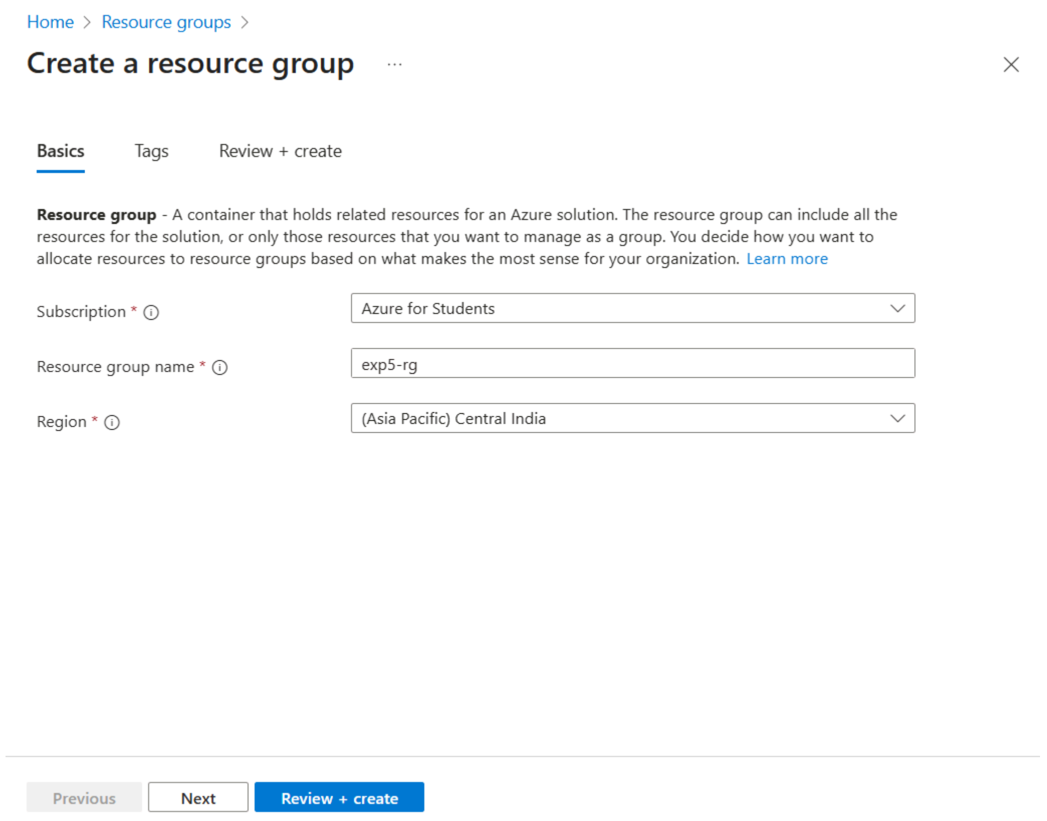

```
C:\Users\dell>ssh azureuser@20.40.40.195
The authenticity of host '20.40.40.195 (20.40.40.195)' can't be established.
ED25519 key fingerprint is SHA256:O7Mhox5P5eomhbO7Y24AMSBnJjged8zobKl5kiJdulM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '20.40.40.195' (ED25519) to the list of known hosts.
azureuser@20.40.40.195's password:
Permission denied, please try again.
azureuser@20.40.40.195's password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.11.0-1018-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Thu Aug 28 03:24:26 UTC 2025

  System load:  0.0               Processes:             122
  Usage of /:   6.7% of 28.02GB   Users logged in:       0
  Memory usage: 4%                IPv4 address for eth0: 10.0.0.4
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

2 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

Exp 5: Create VMs in Public cloud platforms to show load balancing

Step 1: Create a resource group

Create a resource → Virtual machine

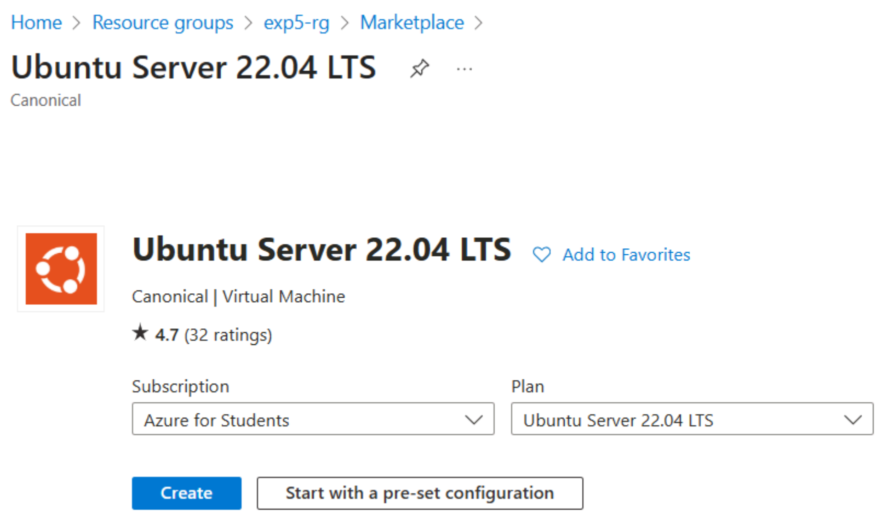
- Resource group: exp5-rg
- VM name: web-1
- Region: same as RG
- Image: **Ubuntu 24.04 LTS**
- Size: available
- Authentication: **Password** (set username azureuser + strong password)



Create a resource group

Resource group - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization. Learn more

Subscription *	Azure for Students
Resource group name *	exp5-rg
Region *	(Asia Pacific) Central India


Home > Resource groups > exp5-rg > Marketplace >

Ubuntu Server 22.04 LTS

Canonical

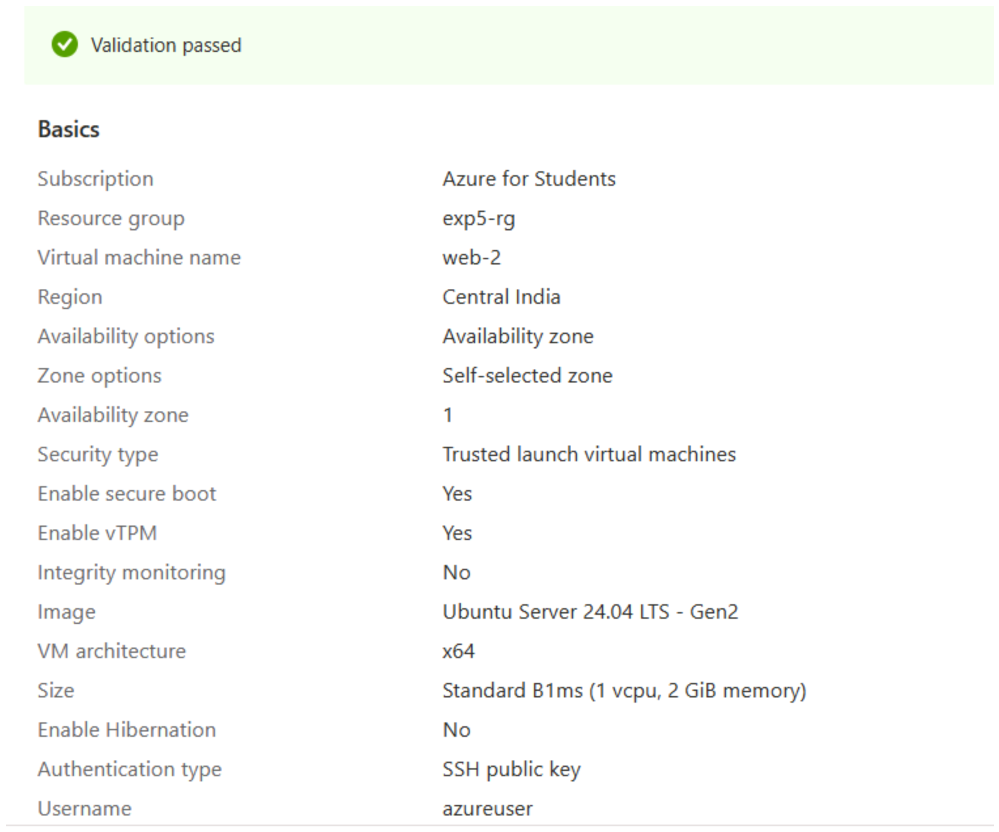
Ubuntu Server 22.04 LTS

Canonical | Virtual Machine

★ 4.7 (32 ratings)

Subscription	Plan
Azure for Students	Ubuntu Server 22.04 LTS

Step 2: Create both VM1 and VM2


Validation passed

Basics

Subscription	Azure for Students
Resource group	exp5-rg
Virtual machine name	web-2
Region	Central India
Availability options	Availability zone
Zone options	Self-selected zone
Availability zone	1
Security type	Trusted launch virtual machines
Enable secure boot	Yes
Enable vTPM	Yes
Integrity monitoring	No
Image	Ubuntu Server 24.04 LTS - Gen2
VM architecture	x64
Size	Standard B1ms (1 vcpu, 2 GiB memory)
Enable Hibernation	No
Authentication type	SSH public key
Username	azureuser

Step 3: Create a web server in VM 1

```
ssh azureuser@4.186.24.129
```

Pwd: SarahRachel@2002

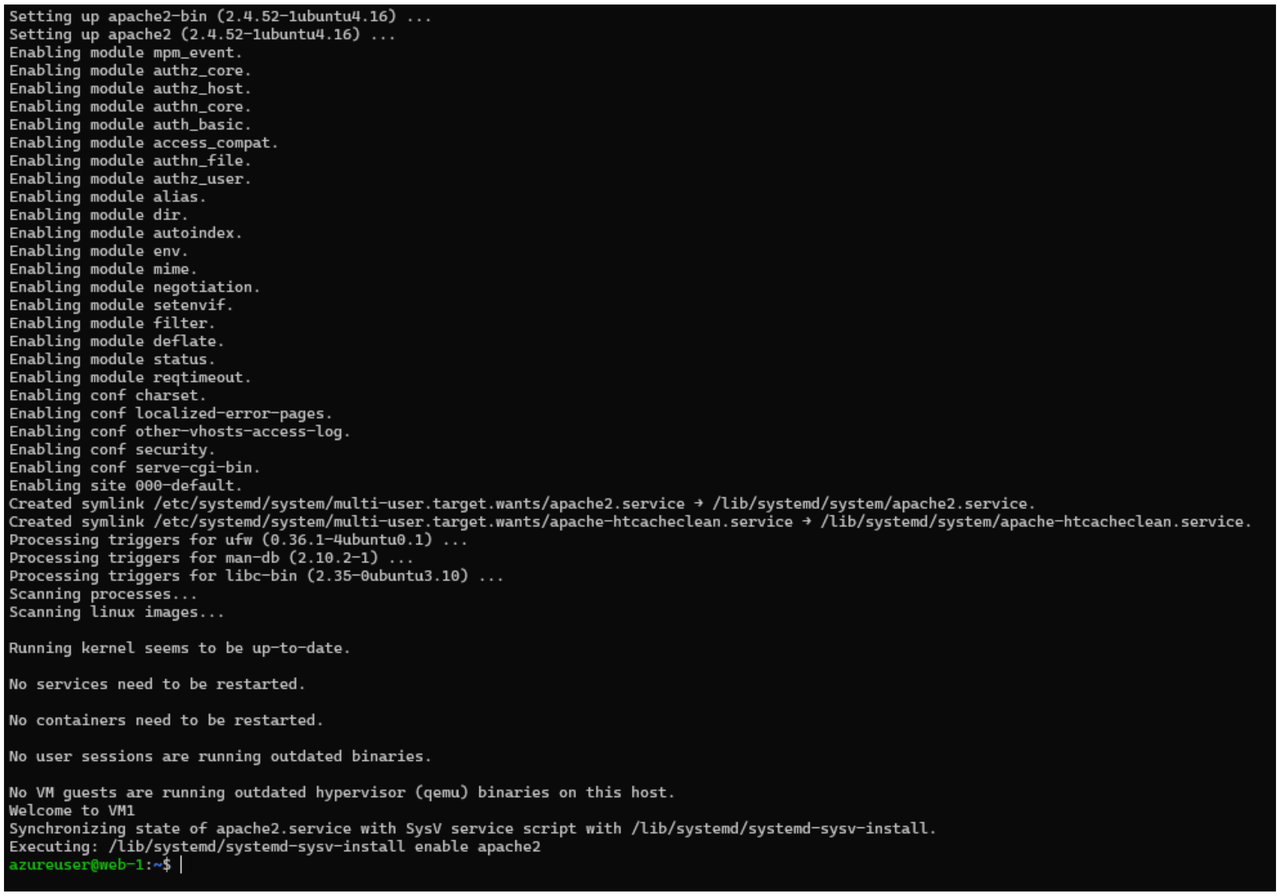
sudo apt update

sudo apt install apache2 -y

echo "Welcome to VM1" | sudo tee /var/www/html/index.html

sudo systemctl enable apache2

sudo systemctl start apache2



```
Setting up apache2-bin (2.4.52-1ubuntu4.16) ...
Setting up apache2 (2.4.52-1ubuntu4.16) ...
Enabling module mpm_event.
Enabling module authz_core.
Enabling module authz_host.
Enabling module authn_core.
Enabling module auth_basic.
Enabling module access_compat.
Enabling module authn_file.
Enabling module authz_user.
Enabling module alias.
Enabling module dir.
Enabling module autoindex.
Enabling module env.
Enabling module mime.
Enabling module negotiation.
Enabling module setenvif.
Enabling module filter.
Enabling module deflate.
Enabling module status.
Enabling module reqtimeout.
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for ufw (0.36.1-4ubuntu0.1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.10) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
Welcome to VM1
Synchronizing state of apache2.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apache2
azureuser@web-1:~$ |
```

Similarly in another terminal for VM2,

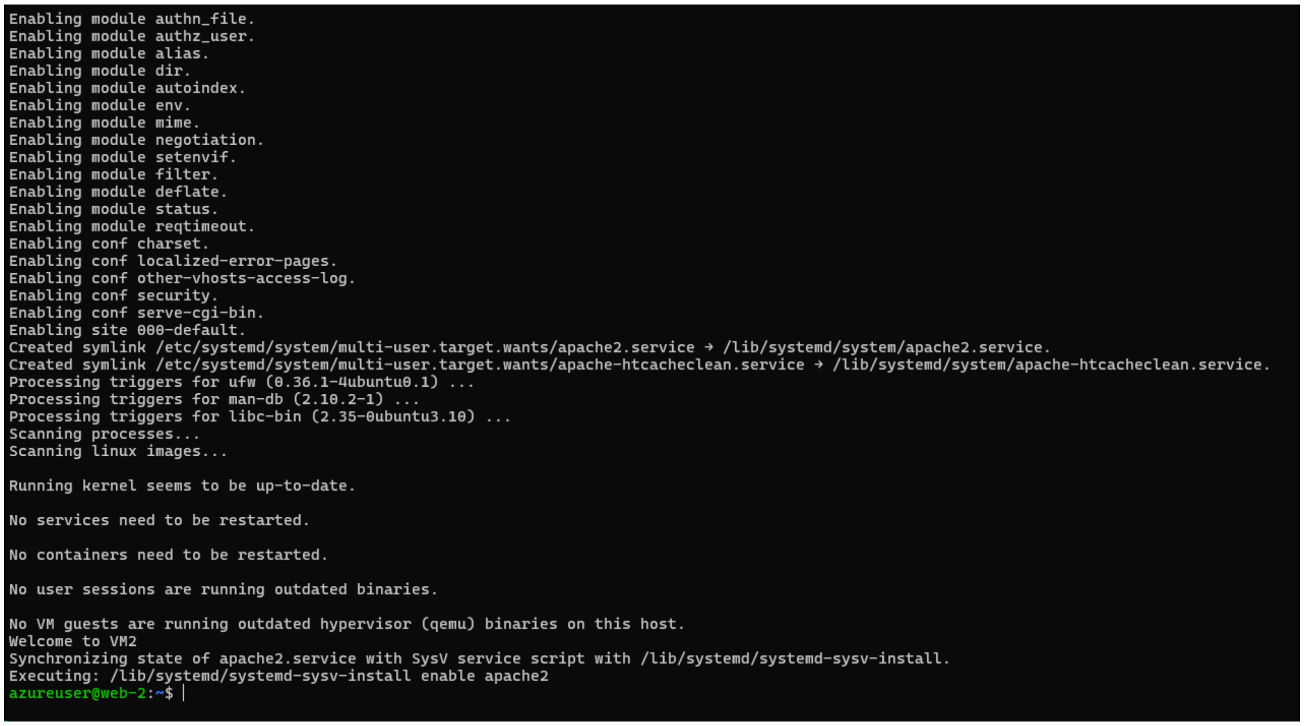
ssh azureuser@20.193.254.29

sudo apt install apache2 -y

echo "Welcome to VM2" | sudo tee /var/www/html/index.html

sudo systemctl enable apache2

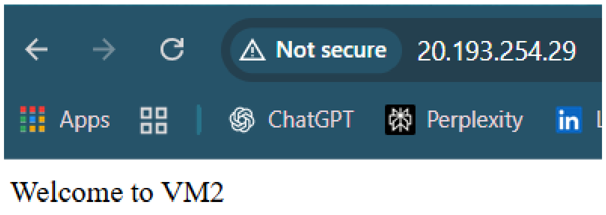
```
sudo systemctl start apache2
```



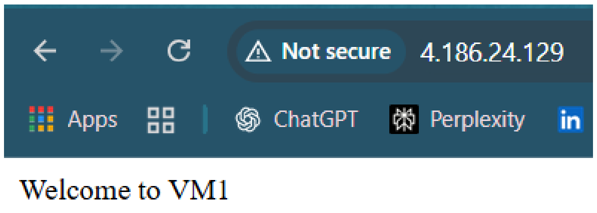
```
Enabling module authn_file.
Enabling module authz_user.
Enabling module alias.
Enabling module dir.
Enabling module autoindex.
Enabling module env.
Enabling module mime.
Enabling module negotiation.
Enabling module setenvif.
Enabling module filter.
Enabling module deflate.
Enabling module status.
Enabling module reqtimeout.
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for ufw (0.36.1-4ubuntu0.1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.10) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
Welcome to VM2
Synchronizing state of apache2.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apache2
azureuser@web-2:~$ |
```

Testing:
Open your browser and enter the Public IP of VM1 → You should see “Welcome to VM1”.
Do the same for VM2’s Public IP → You should see “Welcome to VM2”.
Go to Azure Portal → Resource Group → each VM → Networking → Inbound port rules.
Add a rule like this:

- Source: Any
- Source port ranges: *
- Destination: Any
- Service: HTTP (or custom with port 80)
- Protocol: TCP
- Action: Allow



Welcome to VM2



Welcome to VM1

Now both VMs are ready with Apache.

Step 4: Setup for Load Balancers

Install Nginx on the Load Balancer VM (say VM-LB)

```
sudo apt update

sudo apt install nginx -y
```

Configure Load Balancer

Edit Nginx default config:

```
sudo nano /etc/nginx/sites-available/default
```

Replace the content with:

```
upstream web_backend {
    server <VM1-IP>;
    server <VM2-IP>;
}
server {
    listen 80;
    location / {
        proxy_pass http://web_backend;
    }
}
```

Replace <VM1-IP> and <VM2-IP> with the private IPs of your web servers.

Restart Nginx:

```
sudo systemctl restart nginx

sudo systemctl enable nginx
```

Test Load Balancing: From browser, go to Load Balancer Public IP.

- Refresh multiple times → you should see Welcome to VM1 and Welcome to VM2 alternating.
- VM1 → Apache server
- VM2 → Apache server
- VM-LB → Nginx load balancer

Exp6: Elasticity

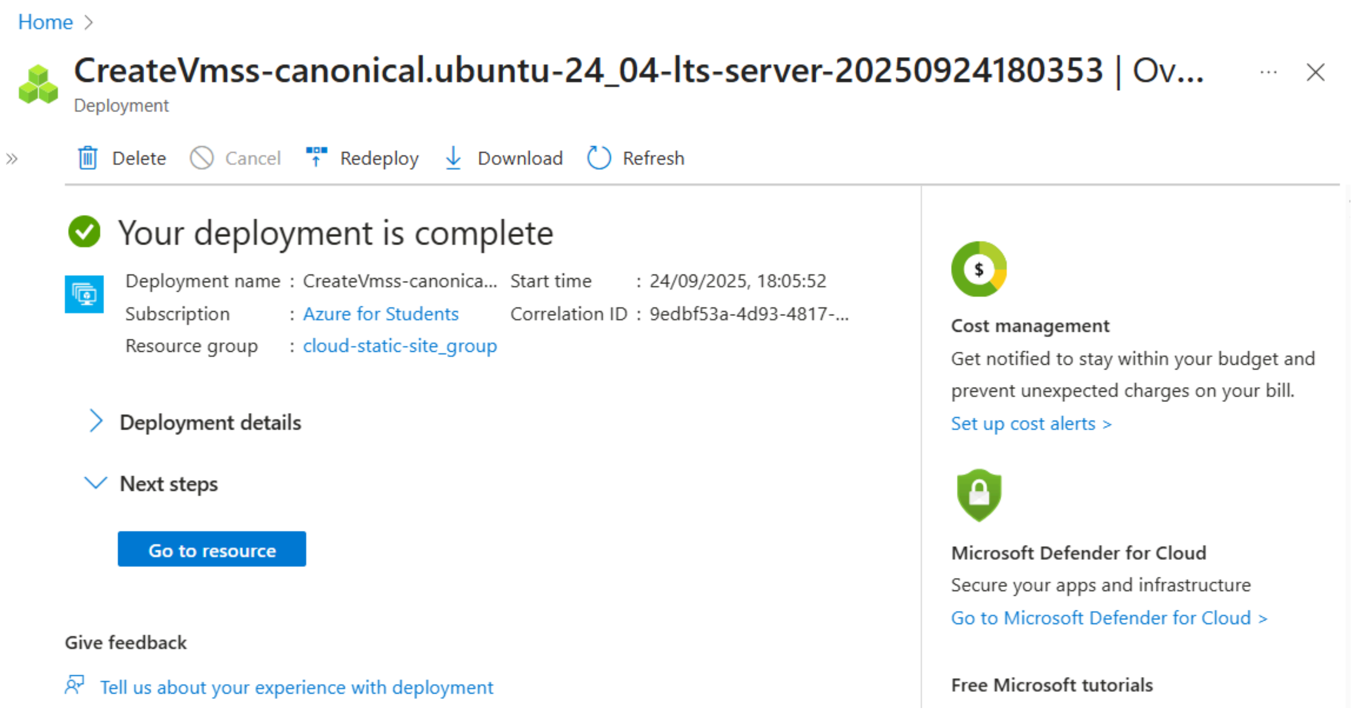
Step 1: Create a Resource Group

1. Go to Azure Portal.
2. Search for **Resource groups → Create**.
   - Subscription: Azure for Students
   - Resource group name: rg-elasticity
   - Region: choose a nearby one (Central India / East US etc.)
3. Click **Review + Create → Create**.

Step 2: Create a Virtual Machine Scale Set (VMSS)

1. In the portal search bar, type **Virtual Machine Scale Sets → Create**.
2. Basics tab:
   - Subscription: Azure for Students
   - Resource group: rg-elasticity
   - VMSS name: vmss-elasticity
   - Region: same as resource group
   - Image: **Ubuntu Server 22.04 LTS**
   - Size: **B1s (1 vCPU, 1 GB RAM)** → ✅ this is included in free tier
   - Authentication: choose **Password** (set username + password you'll remember)

3. Instance details:
    - Initial instance count: 1
    - Scaling: **Enable autoscaling** (we'll configure in Step 3).
4. Networking: allow **HTTP (80)**.
5. Review + Create → Create.



Home >

CreateVmss-canonical.ubuntu-24_04-lts-server-20250924180353 | Ov...

Deployment

Your deployment is complete

Deployment name : CreateVmss-canonica...
Subscription : Azure for Students
Resource group : cloud-static-site_group
Start time : 24/09/2025, 18:05:52
Correlation ID : 9edbf53a-4d93-4817-...

Cost management
Get notified to stay within your budget and prevent unexpected charges on your bill.
Set up cost alerts >

Microsoft Defender for Cloud
Secure your apps and infrastructure
Go to Microsoft Defender for Cloud >

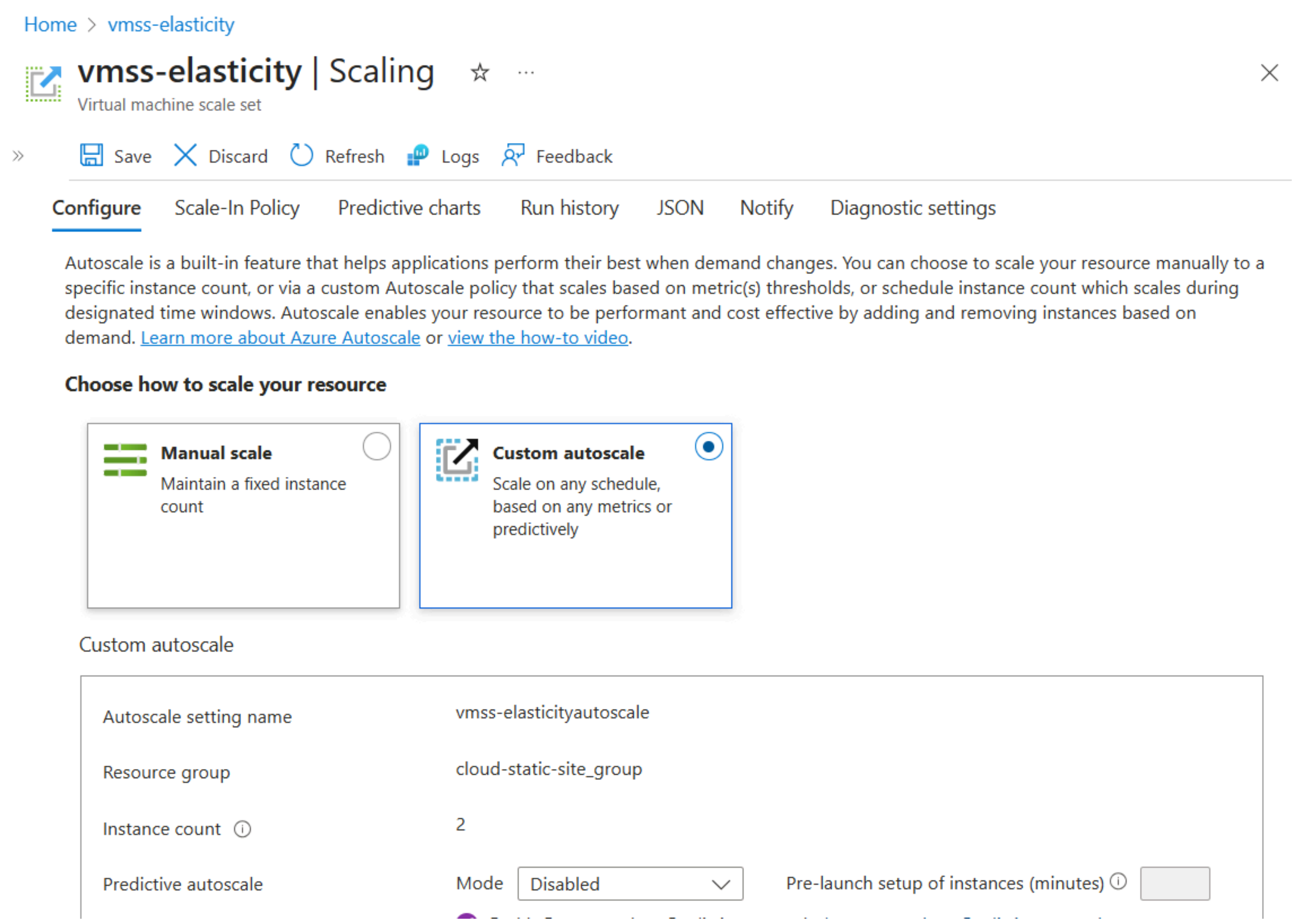
Free Microsoft tutorials

Step 3: Configure Autoscale

1. After deployment, go to your VMSS resource → left menu **Scaling**.
2. Select **Custom autoscale**.
3. Set limits:
    - Minimum = 1

    - Maximum = 3 (don’t go higher, saves credits)
    - Default = 1

4. Add a **Scale-out rule**:
    - Metric: **CPU Percentage**
    - Condition: Greater than **70%**
    - Over the last: **5 minutes**
    - Action: Increase instance count by **1**
5. Add a **Scale-in rule**:
    - Metric: **CPU Percentage**
    - Condition: Less than **30%**
    - Over the last: **10 minutes**
    - Action: Decrease instance count by **1**
6. Save.



Autoscale is a built-in feature that helps applications perform their best when demand changes. You can choose to scale your resource manually to a specific instance count, or via a custom Autoscale policy that scales based on metric(s) thresholds, or schedule instance count which scales during designated time windows. Autoscale enables your resource to be performant and cost effective by adding and removing instances based on demand. Learn more about Azure Autoscale or view the how-to video.

Choose how to scale your resource

Manual scale Maintain a fixed instance count

[✓ **Custom autoscale** Scale on any schedule, based on any metrics or predictively]

Custom autoscale

Autoscale setting name	vmss-elasticityautoscale
Resource group	cloud-static-site_group
Instance count	2
Predictive autoscale	Mode: Disabled; Pre-launch setup of instances (minutes)

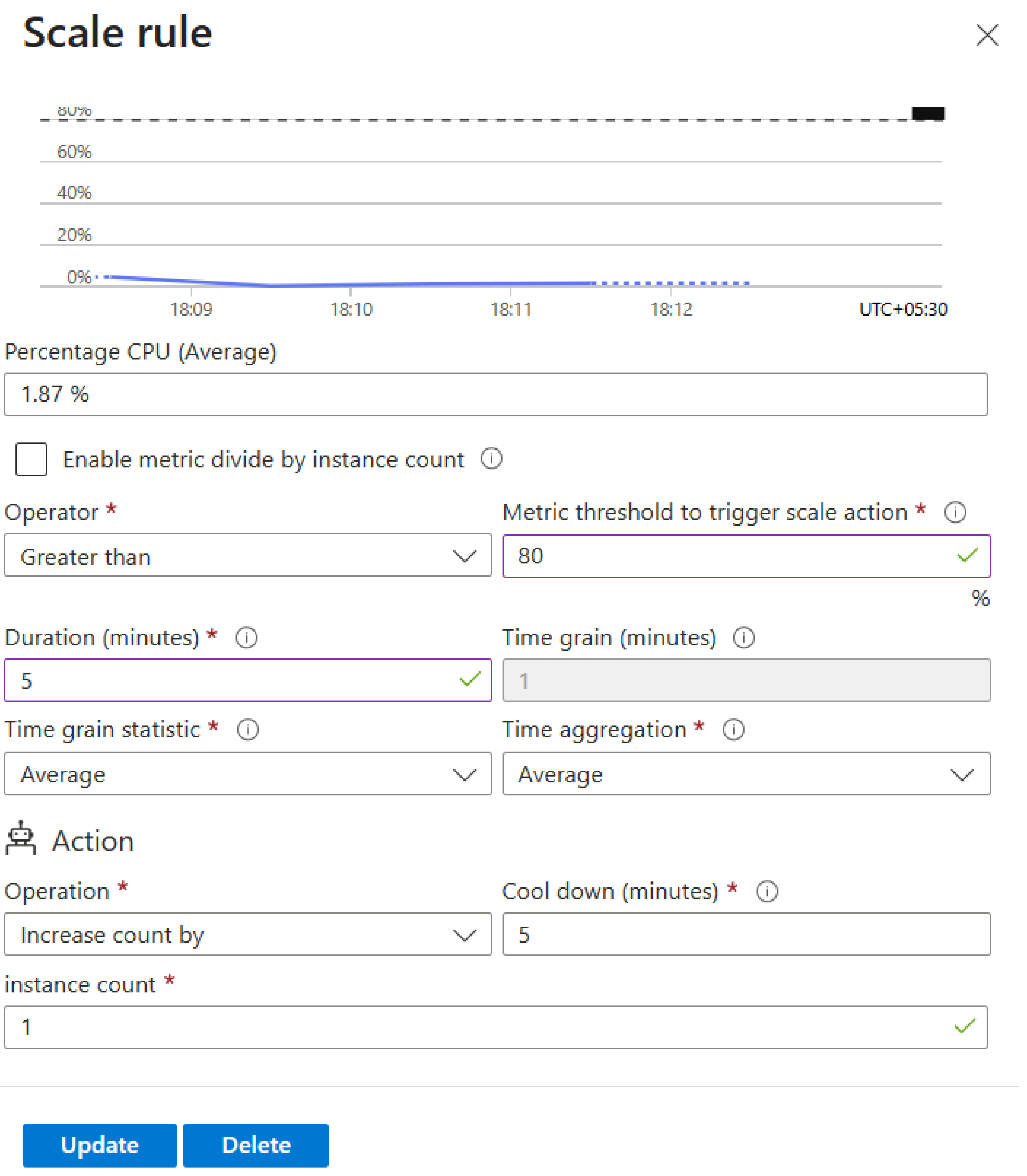
Step 4: **Install via Azure Portal (Custom Script Extension)**

1. Go to your **VM Scale Set** in the Azure Portal.
2. In the left menu, scroll to **Extensions + applications** → click + **Add**.
3. Choose **Custom Script Extension**.

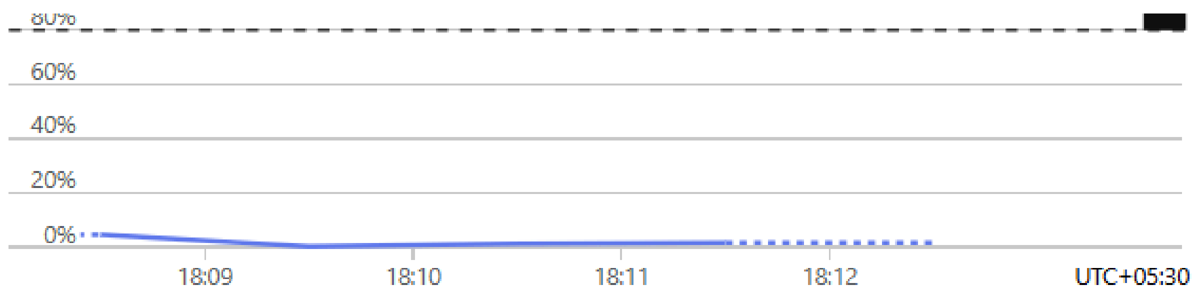
It will ask you to upload a script. Paste this into a file called install-nginx.sh:

```
#!/bin/bash
sudo apt-get update -y
sudo apt-get install -y nginx
sudo systemctl enable nginx
sudo systemctl start nginx

echo "Hello from Azure VM Scale Set!" | sudo tee /var/www/html/index.html
```

4.
5. Upload the script and apply it.
    - This installs **Nginx** (web server) on each VM.
    - The homepage will simply show: *Hello from Azure VM Scale Set!*



Scale rule



Percentage CPU (Average)

1.87 %

Enable metric divide by instance count

Operator *: Greater than

Metric threshold to trigger scale action *: 80 %

Duration (minutes) *: 5

Time grain (minutes): 1

Time grain statistic *: Average

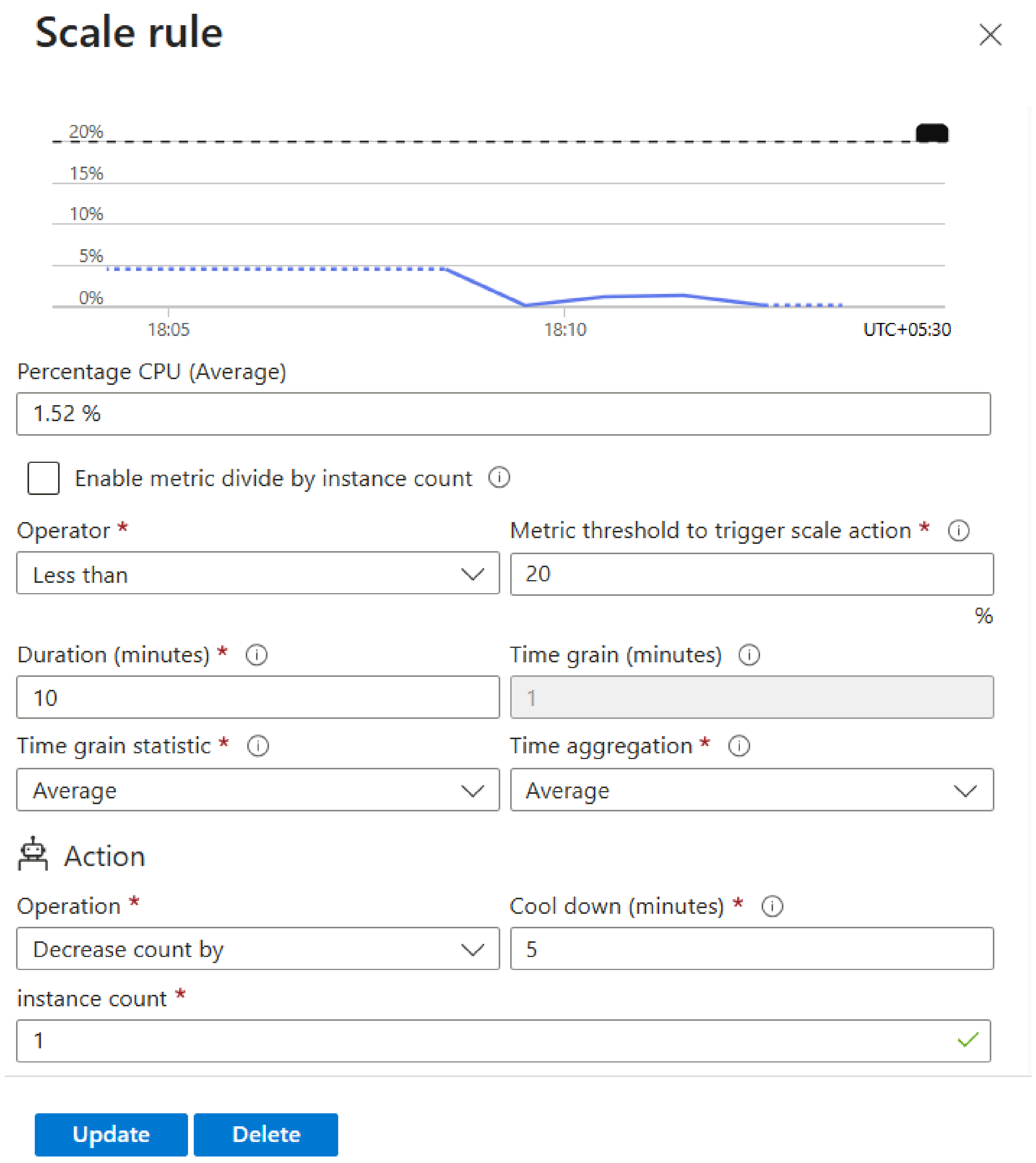
Time aggregation *: Average

Action

Operation *: Increase count by

Cool down (minutes) *: 5

instance count *: 1



Scale rule

Percentage CPU (Average)

1.52 %

☐ Enable metric divide by instance count

Operator *	Metric threshold to trigger scale action *
Less than	20 %
Duration (minutes) *	Time grain (minutes)
10	1
Time grain statistic *	Time aggregation *
Average	Average

Action

Operation *	Cool down (minutes) *
Decrease count by	5

instance count *

1 ✓

Update Delete