

Windows Server Deployment Guide on AWS Cloud

Sou Chanrojame, Orn Pheakdey, Long Neron, Then Sivthean, Le Sreyma

December 20, 2025

Abstract

This document provides a comprehensive step-by-step guide for deploying various Windows Server roles and services on the AWS Cloud infrastructure, including configuration details for EC2 instances, security groups, and storage optimization.

Contents

Prerequisites	4
1 File Server	5
1.1 AWS Configuration	5
1.2 Implementation	5
1.3 Usage	5
2 Proxy Server (Caching, Control Access)	6
2.1 AWS Configuration	6
2.2 Implementation	6
2.3 Usage	6
3 DNS Server	8
3.1 AWS Configuration	8
3.2 Implementation	8
3.3 Usage	8
4 DHCP Server	10
4.1 AWS Configuration	10
4.2 Implementation	10
4.3 Usage	10
5 VPN Server	12
5.1 AWS Configuration	12
5.2 Implementation	12
5.3 Usage	12
6 Terminal Server (Thin Clients)	14
6.1 AWS Configuration	14
6.2 Implementation	14
6.3 Usage	14

7 Web Server	15
7.1 AWS Configuration	15
7.2 Implementation	15
7.3 Usage	15
8 Mail Server	16
8.1 AWS Configuration	16
8.2 Implementation	16
8.3 Usage	17
9 Database Server	19
9.1 AWS Configuration	19
9.2 Implementation	19
9.2.1 PostgreSQL	19
9.2.2 SQL Server	19
9.2.3 MongoDB	20
9.3 Usage	20
10 Backup Server	23
10.1 AWS Configuration	23
10.2 Implementation	23
10.3 Usage	23
11 Load Balancing	25
11.1 AWS Configuration	25
11.2 Implementation Steps	25
11.2.1 Server 1	25
11.2.2 Server 2	25
11.3 Usage	25
12 Failover Cluster	27
12.1 AWS Configuration	27
12.2 Implementation Steps	27
12.3 Common Cluster Types in AWS	27
12.4 Best Practices	28
13 FTP Server	29
13.1 AWS Configuration	29
13.2 Implementation	29
13.3 Usage	31
14 Container (Docker)	32
14.1 AWS Configuration	32
14.2 Implementation	32
14.3 Usage	32
15 Domain Controller	33
15.1 AWS Configuration	33
15.2 Implementation	33
15.3 Usage	33

16 DevSpeedLLM	35
16.1 Configuration	35
16.2 Implementation	35
16.3 Usage	37

Prerequisites

AWS Account Setup

- Active AWS account with appropriate permissions
- VPC configured with public and private subnets
- Security groups properly configured
- Key pairs created for RDP access
- IAM roles for EC2 instances

General Windows Server Launch Steps

1. Navigate to EC2 Dashboard in AWS Console
2. Click "Launch Instance"
3. Select Windows Server AMI (2019/2022 recommended)
4. Choose instance type based on workload
5. Configure instance details (VPC, subnet, IAM role)
6. Add storage as needed
7. Configure security groups
8. Review and launch with key pair

1 File Server

1.1 AWS Configuration

Instance Name: File Server

Instance Type: t3.medium or larger

Storage: EBS volumes with provisioned IOPS for performance

Security Group Ports: tcp 445 (SMB), tcp 3389 (RDP)

1.2 Implementation

```
1 Install-WindowsFeature -Name FS-FileServer
2 New-Item -Path "C:\FileShare" -ItemType Directory -Force
3 New-SmbShare -Name "PublicShare" -Path "C:\FileShare" -FullAccess "Everyone"
4 echo hello > C:\FileShare\hello.txt
5 Write-Host "File Server setup complete."
```

1.3 Usage

- connect to Proxy Server or other windows server
- open file explorer and \\<public dns of File Server>
- enter credential

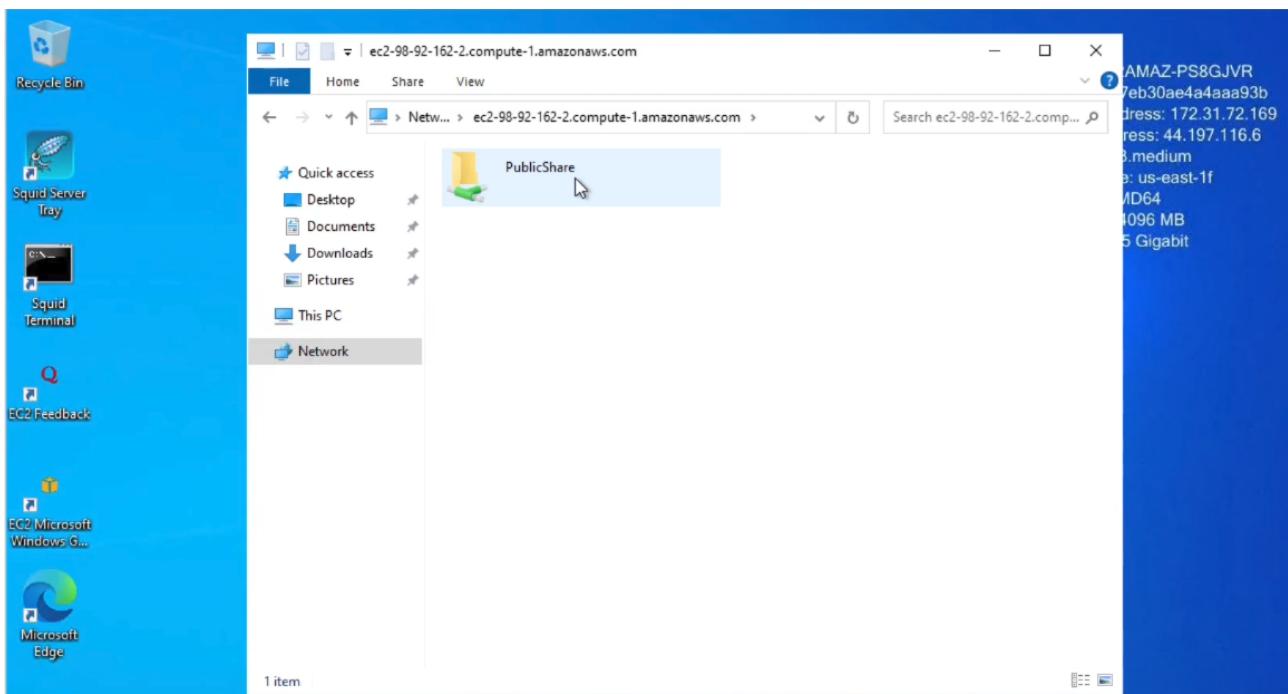


Figure 1: Successfully connected to file server

2 Proxy Server (Caching, Control Access)

2.1 AWS Configuration

Instance Name: Proxy Server

Instance Type: t3.medium

Security Group Ports: tcp 3128, tcp 3389 (RDP)

2.2 Implementation

```
1 Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::  
    SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex  
    ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/  
    install.ps1'))  
2 choco install squid -y  
3 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/  
    refs/heads/main/config/squid.conf -OutFile "C:\Squid\etc\squid\squid.conf"  
4 Restart-Service squidsvr  
5 Write-Host "Proxy Server setup complete."
```

2.3 Usage

- connect to File Server or other windows server
- change proxy address to <private ip of Proxy Server> with port 3128
- open browser
- visit youtube.com => allow
- visit facebook.com => blocked

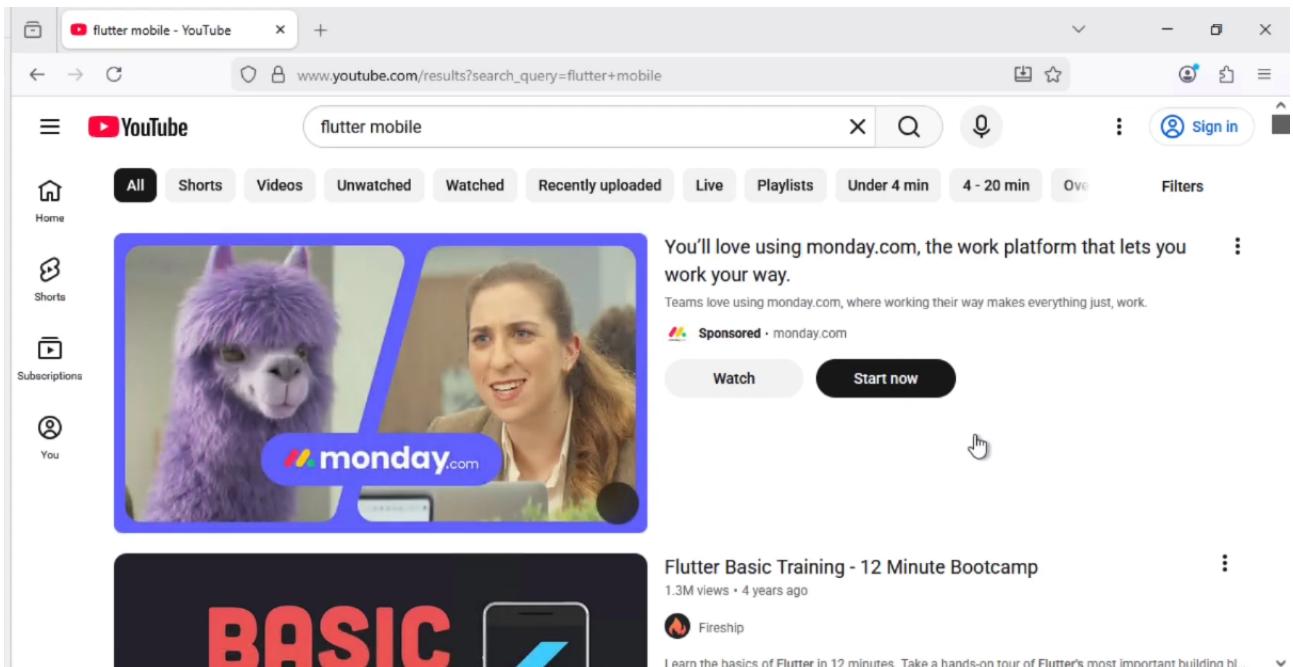


Figure 2: Allow access to youtube

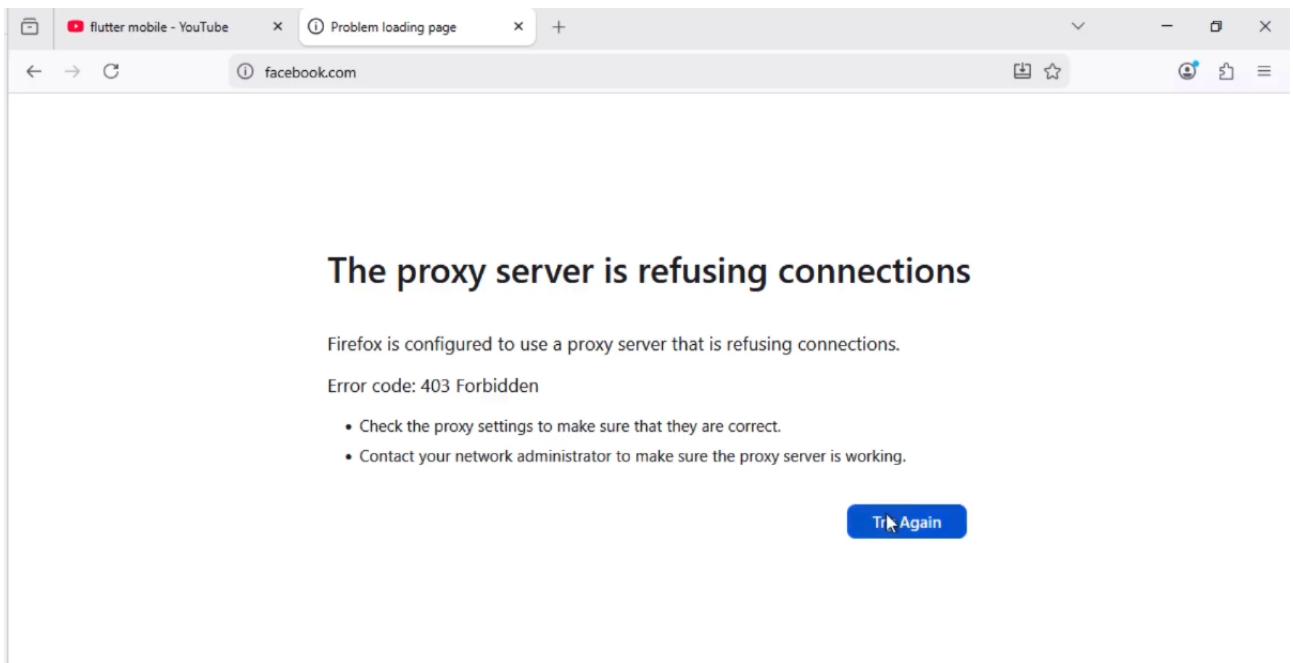


Figure 3: Block access to facebook

3 DNS Server

3.1 AWS Configuration

Instance Name: DNS Server

Instance Type: t3.medium

Security Group Ports: tcp/udp 53, tcp 3389 (RDP)

3.2 Implementation

```
1 Install-WindowsFeature DNS -IncludeManagementTools
2
3 Add-DnsServerPrimaryZone -Name "example.internal" -ZoneFile "example.internal.dns"
4 Add-DnsServerPrimaryZone -Name "devspeed.com" -ZoneFile "devspeed.com.dns"
5
6 Add-DnsServerResourceRecordA -Name 'server1' -ZoneName 'example.internal' -IPv4Address '10.0.1.10'
7 Add-DnsServerResourceRecordA -Name 'console' -ZoneName 'devspeed.com' -IPv4Address '192.168.1.11'
8 Add-DnsServerResourceRecordA -Name 'go' -ZoneName 'devspeed.com' -IPv4Address '192.168.1.29'
9 Add-DnsServerResourceRecordA -Name 'blog' -ZoneName 'devspeed.com' -IPv4Address '192.168.1.30'
10 Add-DnsServerResourceRecordA -Name 'shop' -ZoneName 'devspeed.com' -IPv4Address '192.168.1.31'
11 Add-DnsServerResourceRecordA -Name 'support' -ZoneName 'devspeed.com' -IPv4Address '192.168.1.32'
12 Add-DnsServerResourceRecordA -Name 'mail' -ZoneName 'devspeed.com' -IPv4Address '192.168.1.33'
13 Add-DnsServerResourceRecordA -Name 'www' -ZoneName 'devspeed.com' -IPv4Address '192.168.1.34'
14 Add-DnsServerResourceRecordA -Name 'www2' -ZoneName 'devspeed.com' -IPv4Address '192.168.1.100'
15
16 # For example, forwarding to Google DNS:
17 # Add-DnsServerForwarder -IPAddress "8.8.8.8" -PassThru
18 # Or forwarding to AWS VPC DNS (recommended for EC2). This is the Amazon-provided DNS resolver for VPCs.
19 # Add-DnsServerForwarder -IPAddress "169.254.169.253" -PassThru
20
21 New-NetFirewallRule -DisplayName "Allow DNS TCP 53" -Direction Inbound -Protocol TCP -LocalPort 53 -Action Allow
22 New-NetFirewallRule -DisplayName "Allow DNS UDP 53" -Direction Inbound -Protocol UDP -LocalPort 53 -Action Allow
```

3.3 Usage

- open Terminal
- dog www.devspeed.com '@<public ip of dns server>'
- dog go.devspeed.com '@<public ip of dns server>'
- nslookup shop.devspeed.com '<public ip of dns server>'
- nslookup mail.devspeed.com '<public ip of dns server>'

```
[jame Jame-Linux] - [~] - [2025-12-12 02:46:07]
[0] <> dog www.devspeed.com @44.222.65.29
A www.devspeed.com. 1h00m00s 192.168.1.34
[jame Jame-Linux] - [~] - [2025-12-12 02:46:20]
[0] <> dog go.devspeed.com @44.222.65.29
A go.devspeed.com. 1h00m00s 192.168.1.29
[jame Jame-Linux] - [~] - [2025-12-12 02:46:41]
[0] <> nslookup shop.devspeed.com 44.222.65.29
Server: 44.222.65.29
Address: 44.222.65.29#53

Name: shop.devspeed.com
Address: 192.168.1.31
```

Figure 4: Dns server response with respective IP address

4 DHCP Server

4.1 AWS Configuration

Instance Name: DHCP Server

Instance Type: t3.medium

Security Group Ports: udp 67, udp 68, tcp 3389 (RDP)

Note

AWS VPC provides DHCP by default; custom DHCP server is optional.

Is the dhcp server in ec2 instance works in AWS, because ec2 instance already used VPC DHCP?

No — a normal DHCP server inside an EC2 instance does NOT work for assigning IPs to other EC2 instances.

Because:

- 👉 AWS VPC already provides DHCP
- 👉 DHCP broadcast is blocked in AWS

4.2 Implementation

```
1 # Install DHCP Role
2 Install-WindowsFeature DHCP -IncludeManagementTools
3
4 # If in a domain (skip if standalone)
5 # Add-DhcpServerInDC -DnsName "dhcpserver.example.internal" -IpAddress "10.0.1.10"
6
7 # Create scope
8 Add-DhcpServerv4Scope ` 
9     -Name "MainScope" ` 
10    -StartRange 10.0.1.50 ` 
11    -EndRange 10.0.1.150 ` 
12    -SubnetMask 255.255.255.0 ` 
13    -State Active
14
15 # Configure DHCP options
16 Set-DhcpServerv4OptionValue -ScopeId 10.0.1.0 -Router 10.0.1.1
17 Set-DhcpServerv4OptionValue -ScopeId 10.0.1.0 -DnsServer 10.0.1.10
18 Set-DhcpServerv4OptionValue -ScopeId 10.0.1.0 -DnsDomain "example.internal"
19
20 New-NetFirewallRule -DisplayName "Allow DHCP UDP 67" -Direction Inbound -Protocol UDP -
21 LocalPort 67 -Action Allow
22 New-NetFirewallRule -DisplayName "Allow DHCP UDP 68" -Direction Inbound -Protocol UDP -
23 LocalPort 68 -Action Allow
```

4.3 Usage

- AWS VPC DHCP is used in EC2 instance

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like EC2, Instances, Images, and Elastic Block Store. The main area displays a table of instances. One instance, 'ProxyServerDHCPServer' (ID: i-019e1b125793098c0), is selected and highlighted with a blue border. The table columns include Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. Below the table, a detailed view for the selected instance is shown, with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. The Details tab is active, showing the Instance summary. It includes fields for Instance ID (i-019e1b125793098c0), Public IPv4 address (44.222.187.240), Private IPv4 addresses (172.31.73.150), Public DNS (ec2-44-222-187-240.compute-1.amazonaws.com), and Instance state (Running). There are also links to open the address for each.

Figure 5: A dhcp server instance

5 VPN Server

5.1 AWS Configuration

Instance Name: VPN Server

Instance Type: t3.small to t3.medium

Security Group Ports: udp 1194, tcp 3389 (RDP)

Elastic IP: Required for consistent endpoint

5.2 Implementation

```
1 Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::  
2 SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex  
3 ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/  
4 install.ps1'))  
5 choco install openvpn -y --package-parameters=" /AddToDesktop /Gui /GuiOnLogon /EasyRsa  
6 /DcoDriver /TapDriver /WintunDriver /OpenSSL /Service "  
7 choco install caddy 7zip -y  
8  
9 $cfg = "C:\Program Files\OpenVPN\config"  
10  
11 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/  
12     refs/heads/main/config/openvpn/server/dh.pem -OutFile "$cfg\dh.pem"  
13 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/  
14     refs/heads/main/config/openvpn/server/server.crt -OutFile "$cfg\server.crt"  
15 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/  
16     refs/heads/main/config/openvpn/server/server.key -OutFile "$cfg\server.key"  
17 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/  
18     refs/heads/main/config/openvpn/server/ca.crt -OutFile "$cfg\ca.crt"  
19 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/  
20     refs/heads/main/config/openvpn/server/server.ovpn -OutFile "$cfg\server.ovpn"  
21  
22 Set-Service -Name OpenVPNService -StartupType Automatic  
23 Start-Service -Name OpenVPNService  
24  
25 New-NetFirewallRule -DisplayName "OpenVPN" -Direction Inbound -Protocol UDP -LocalPort  
26     1194 -Action Allow  
27 New-NetFirewallRule -DisplayName "ShareFileOpenVPN" -Direction Inbound -Protocol TCP -  
28     LocalPort 80 -Action Allow  
29  
30 # don't know why it doesn't work. (The server doesn't run)  
31 # sleep -Seconds 5 # still not working  
32 # Start-Process "C:\Program Files\OpenVPN\bin\openvpn-gui.exe"  
33  
34 # # for client to openvpn server  
35 # $cfg = "C:\Program Files\OpenVPN\config"  
36 # Invoke-WebRequest <url> "$cfg\client1.ovpn"  
37 # Invoke-WebRequest <url> "$cfg\ca.crt"  
38 # Invoke-WebRequest <url> "$cfg\client1.crt"  
39 # Invoke-WebRequest <url> "$cfg\client1.key"
```

5.3 Usage

- connect to VPN Server
- Open OpenVPN GUI to start the server
- connect to File Server

- change YOUR_PUBLIC_IP in C:\Program Files\OpenVPN\config\client1.ovpn to public ip of the VPN Server
- Open OpenVPN GUI to connect to the server
- Open powershell and type ipconfig and will see something like:

```

1 Unknown adapter OpenVPN Data Channel Offload:
2
3   Connection-specific DNS Suffix  . :
4     Link-local IPv6 Address . . . . . : fe80::f729:5f67:58f2:7253%17
5     IPv4 Address . . . . . : 10.8.0.6
6     Subnet Mask . . . . . : 255.255.255.252
7     Default Gateway . . . . . :

```

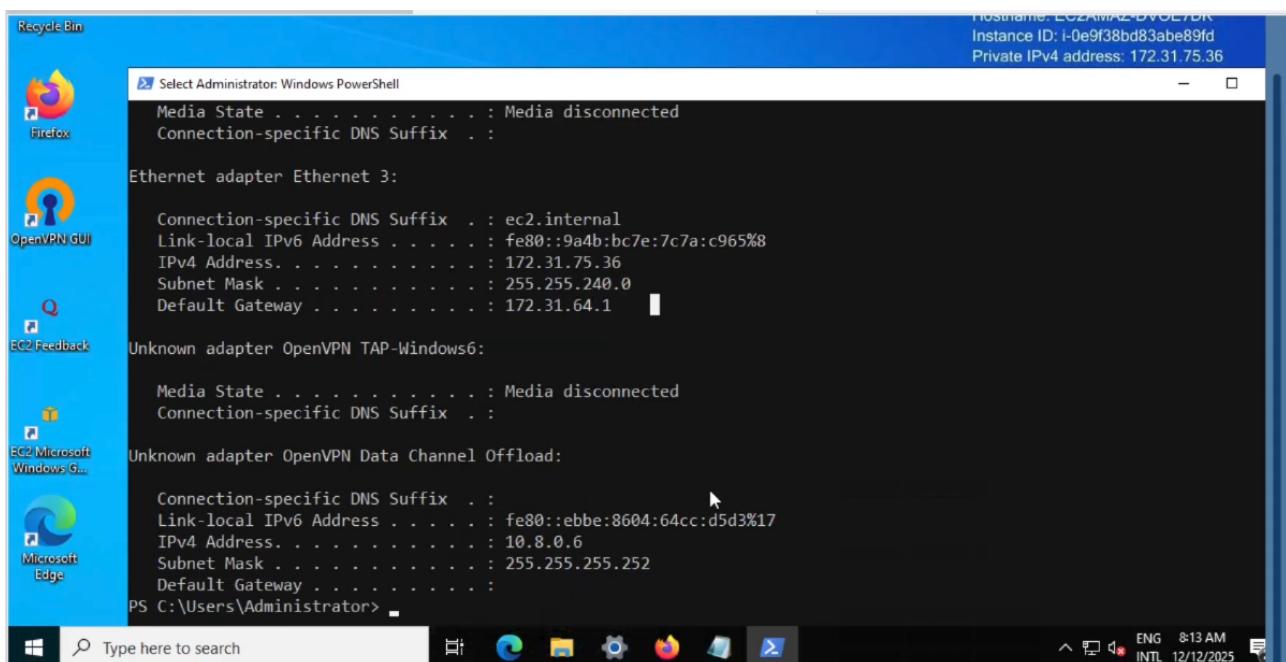


Figure 6: vpn client successfully connect to VPN server

6 Terminal Server (Thin Clients)

6.1 AWS Configuration

Instance Name: Terminal Server

Instance Type: t3.medium

Security Group Ports: tcp 3389 (RDP)

6.2 Implementation

```
1 # Install Remote Desktop Session Host Role
2 Install-WindowsFeature RDS-RD-Server
3 # Enable Multiple Sessions
4 Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server" -Name
   fSingleSessionPerUser -Value 0
5 # Allow RDP Firewall Rule
6 Enable-NetFirewallRule -DisplayGroup "Remote Desktop"
7 Write-Host "Terminal Server (RDS Session Host) setup complete."
```

6.3 Usage

- connect to the server with RDP

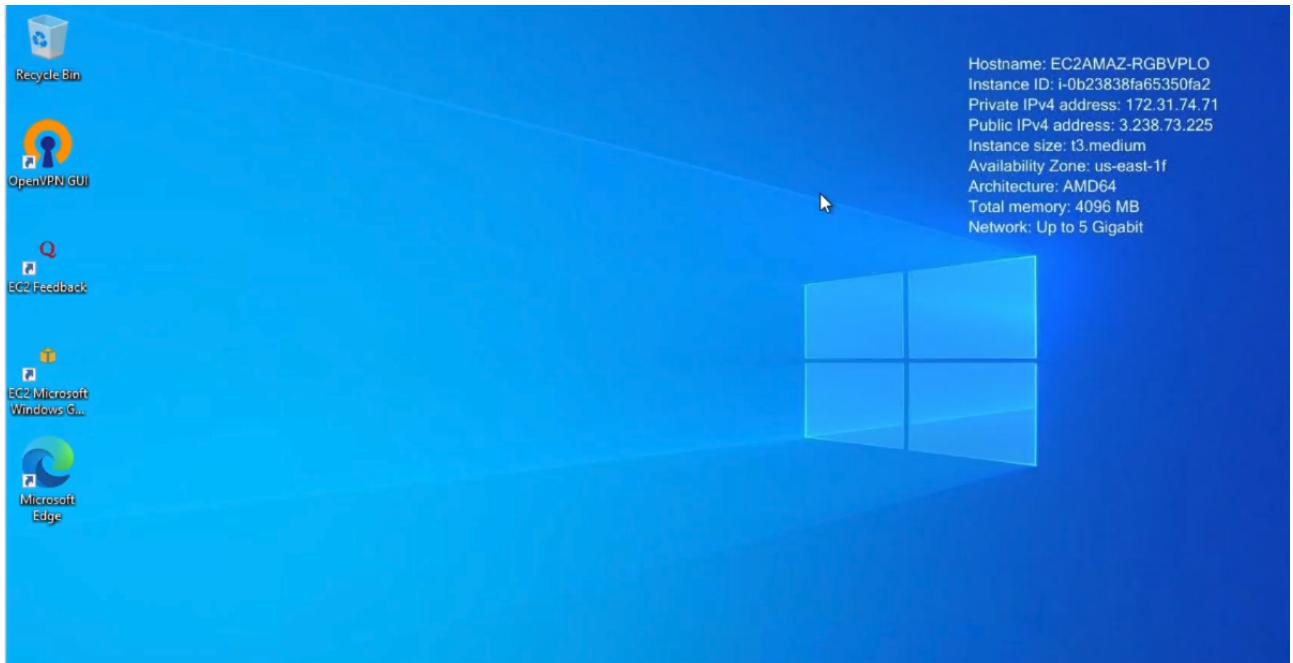


Figure 7: Successfully connect to terminal server

7 Web Server

7.1 AWS Configuration

Instance Name: Web Server

Instance Type: t3.medium

Security Group Ports: tcp 80 (HTTP), tcp 443 (HTTPS), tcp 3389 (RDP)

7.2 Implementation

```
8 Install-WindowsFeature -Name Web-Server -IncludeManagementTools
9 # Install-WindowsFeature -Name Web-Asp-Net45, Web-Net-Ext45
10 # Install-WindowsFeature -Name Web-Mgmt-Console
11 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/
   refs/heads/main/static/devspeed.html -OutFile "C:\inetpub\wwwroot\index.html"
```

7.3 Usage

- visit public dns of the server

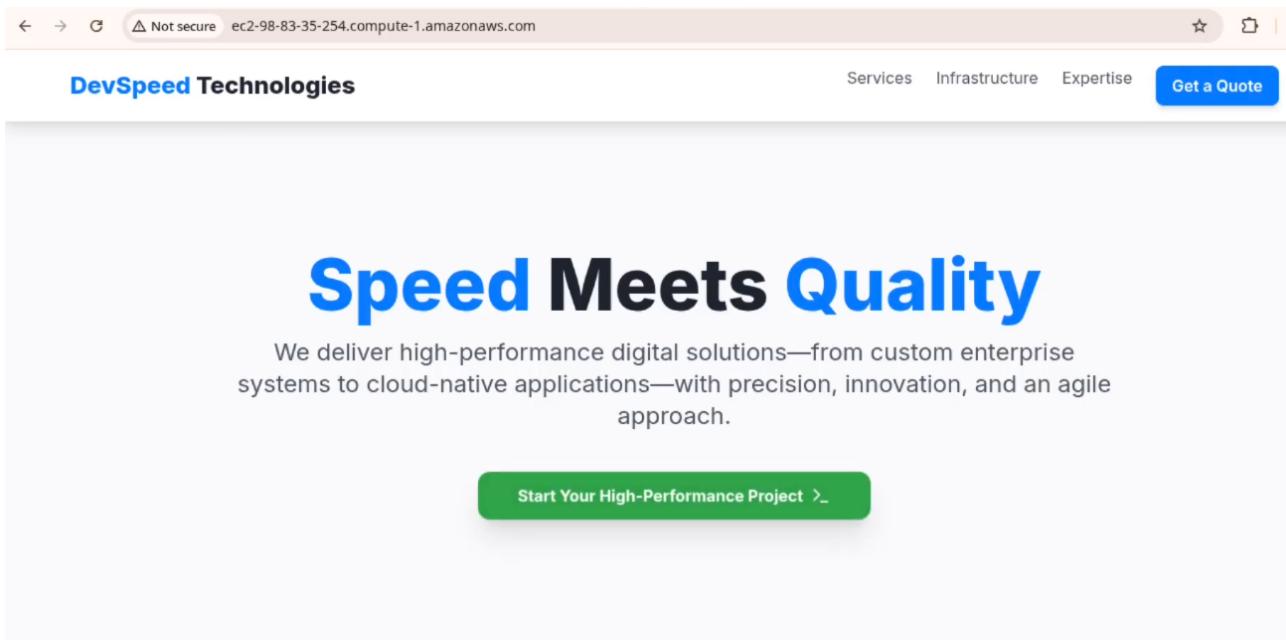


Figure 8: visiting the web page

8 Mail Server

8.1 AWS Configuration

Instance Type: t3.medium

Security Group Ports: 25 (SMTP), 110 (POP3), 143 (IMAP), 587 (Submission), 993 (IMAPS), 995 (POP3S)

Elastic IP: Required

Note: AWS blocks port 25 by default; request removal.

8.2 Implementation

```
1 # mail server
2 # - apache james (smtp server [port: 25], imap server [port: 143])
3 # - swaks (smtp client)
4 # - thunderbird (email client)
5
6 Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
7 choco install openjdk strawberryperl thunderbird notepadplusplus telnet -y
8 Invoke-WebRequest https://www.jetmore.org/john/code/swaks/files/swaks-20240103.0/swaks -OutFile swaks.pl
9 Invoke-WebRequest https://dlcdn.apache.org/james/server/3.9.0/james-server-spring-app-3.9.0-app.zip -OutFile james-server-spring-app-3.9.0-app.zip
10 Expand-Archive james-server-spring-app-3.9.0-app.zip -DestinationPath james-server-spring-app-3.9.0-app
11
12 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/refs/heads/main/config/smtpserver.xml -OutFile "james-server-spring-app-3.9.0-app\james-server-spring-app-3.9.0\conf\smtpserver.xml"
13 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/refs/heads/main/config/imapserver.xml -OutFile "james-server-spring-app-3.9.0-app\james-server-spring-app-3.9.0\conf\imapserver.xml"
14
15 Import-Module $env:ChocolateyInstall\helpers\chocolateyProfile.psm1
16 refreshenv
17
18 # cd "james-server-spring-app-3.9.0-app\james-server-spring-app-3.9.0\bin\
19
20 $james = "C:\Windows\System32\james-server-spring-app-3.9.0-app\james-server-spring-app-3.9.0\bin\james.bat"
21 $james_cli = "C:\Windows\System32\james-server-spring-app-3.9.0-app\james-server-spring-app-3.9.0\bin\james-cli.ps1"
22
23 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/refs/heads/main/scripts/james-cli.ps1 -OutFile $james_cli
24
25 function Wait-JamesReady {
26     param(
27         [int]$Port = 9999,
28         [string]$Url = "localhost",
29         [int]$TimeoutSeconds = 120
30     )
31
32     $start = Get-Date
33
34     Write-Host "Waiting for James server to open JMX on port $Port..."
35 }
```

```

36 while ((Get-Date) -lt $start.AddSeconds($TimeoutSeconds)) {
37     if ((Test-NetConnection -ComputerName $Url -Port $Port -WarningAction
38         SilentlyContinue).TcpTestSucceeded) {
39         Write-Host "James server is ready!"
40         return
41     }
42     Start-Sleep -Seconds 2
43 }
44
45 throw "Timeout: James server did not open JMX port $Port"
46 }
47
48 & $james install
49 & $james start
50
51 # need to wait james server completely ready
52 Wait-JamesReady
53
54 & $james_cli -username james-admin -password changeme adddomain example.local
55 & $james_cli -username james-admin -password changeme adduser mike@example.local mike
56 & $james_cli -username james-admin -password changeme adduser joe@example.local joe
57 & $james_cli -username james-admin -password changeme adduser leng@example.local leng
58 & $james_cli -username james-admin -password changeme adduser jame@example.local jame
59
60 New-NetFirewallRule -DisplayName "Allow smtp 25" -Direction Inbound -Protocol TCP -
61     LocalPort 25 -Action Allow
62 New-NetFirewallRule -DisplayName "Allow imap 143" -Direction Inbound -Protocol TCP -
63     LocalPort 143 -Action Allow

```

8.3 Usage

- Open Thunderbird
- Username: jame@example.local
- Password: jame
- Username: mike@example.local
- Password: mike
- IMAP
 - Hostname: <public-ip-of-mail-server>
 - Port: 143
- SMTP
 - Hostname: <public-ip-of-mail-server>
 - Port: 25

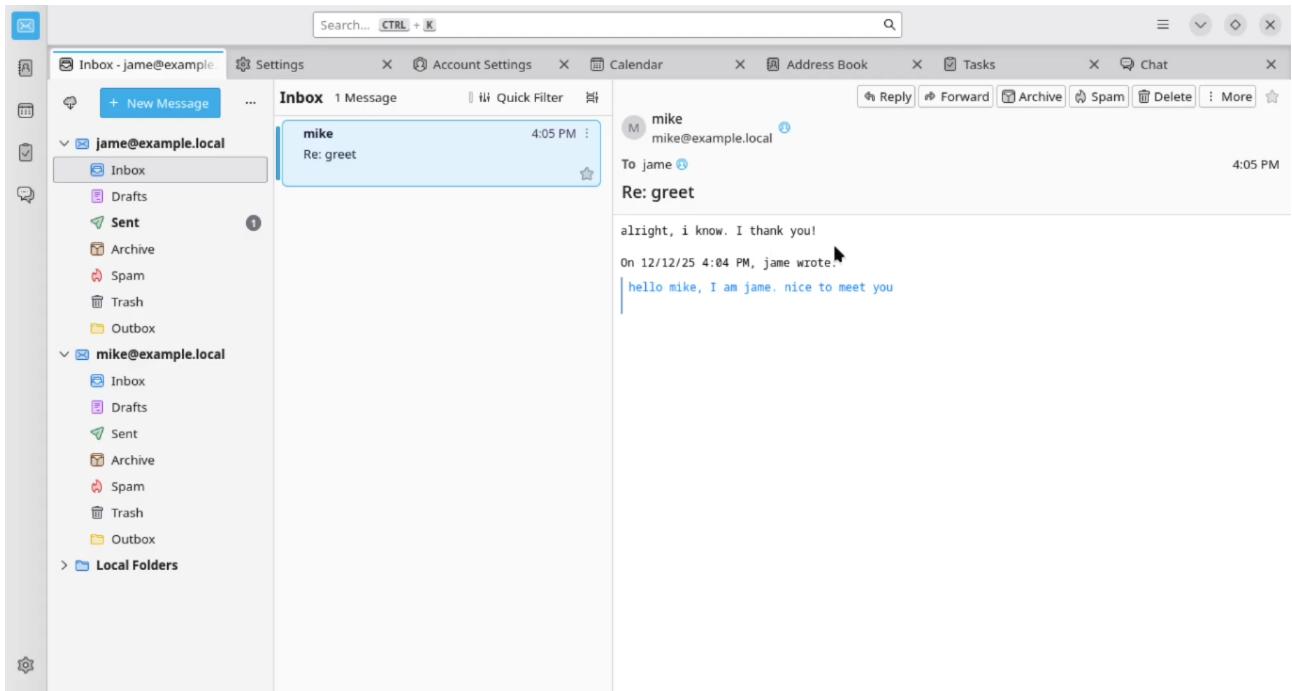


Figure 9: Two users sending messages to each other

9 Database Server

9.1 AWS Configuration

Instance Name: Database Server

Instance Type: t3.medium or larger (memory-optimized)

Storage: EBS with provisioned IOPS or io2

Security Group Ports:

- PostgreSQL: tcp 5432
- SQL Server: tcp 1433
- MongoDB: tcp 27017
- RDP: tcp 3389

9.2 Implementation

9.2.1 PostgreSQL

```
1 # postgresql
2 Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::
3     SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex
4     ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/
5     install.ps1'))
6 # username postgres
7 choco install postgresql11 --params '/Password:test /AllowRemote' -y
8 Restart-Service postgresql-x64-11
9 New-NetFirewallRule -DisplayName "Allow PostgreSQL 5432" ` 
10    -Direction Inbound ` 
11    -Protocol TCP ` 
12    -LocalPort 5432 ` 
13    -Action Allow
```

9.2.2 SQL Server

```
1 # sql server
2 Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::
3     SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex
4     ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/
5     install.ps1'))
6 choco install mssqlserver2014express -y
7 choco install sqlserver-cmdlineutils -y
8
9 # Get access to SqlWmiManagement DLL on the machine with SQL
10 # we are on, which is where SQL Server was installed.
11 # Note: This is installed in the GAC by SQL Server Setup.
12
13 # Load the WMI assembly
14 [System.Reflection.Assembly]::LoadWithPartialName('Microsoft.SqlServer.SqlWmiManagement')
15     ) | Out-Null
16
17 # Connect to local SQL Server machine
18 $wmi = New-Object 'Microsoft.SqlServer.Management.Smo.Wmi.ManagedComputer' 'localhost'
19
20 # Get TCP protocol for SQLEXPRESS
21 $tcp = $wmi.ServerInstances['SQLEXPRESS'].ServerProtocols['Tcp']
```

```

18 $tcp.IsEnabled = $true
19
20 # Disable dynamic ports on each IP entry
21 foreach ($ip in $tcp.IPAddresses) {
22     $ip.IPAddressProperties["TcpDynamicPorts"].Value = ""
23 }
24
25 # Set static port 1433 on ALL IPs (including IPAll)
26 foreach ($ip in $tcp.IPAddresses) {
27     $ip.IPAddressProperties["TcpPort"].Value = "1433"
28 }
29
30 # Apply configuration
31 $tcp.Alter()
32
33 # You need to restart SQL Server for the change to persist
34 # -Force takes care of any dependent services, like SQL Agent.
35 # Note: If the instance is named, replace MSSQLSERVER with MSSQL$ followed by
36 # the name of the instance (e.g., MSSQL$MYINSTANCE)
37
38 Restart-Service -Name 'MSSQL$SQLEXPRESS' -Force
39
40 & "C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\110\Tools\Binn\SQLCMD.EXE" -S
    .\SQLEXPRESS -Q "ALTER LOGIN sa ENABLE; ALTER LOGIN sa WITH PASSWORD = "
    'mypassword@2025';
41 Set-ItemProperty -Path "HKLM:\SOFTWARE\Microsoft\Microsoft SQL Server\MSSQL12.SQLEXPRESS
    \MSSQLServer" -Name LoginMode -Value 2
42 Restart-Service -Name 'MSSQL$SQLEXPRESS' -Force
43
44 New-NetFirewallRule -DisplayName "Allow SQL Server 1433" ` 
    -Direction Inbound ` 
    -Protocol TCP ` 
    -LocalPort 1433 ` 
    -Action Allow
45
46
47
48

```

9.2.3 MongoDB

```

1 # mongodb
2 Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::
3     SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex
4     ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/
5     install.ps1'))
6 choco install mongodb -y
7 choco install mongodb-shell -y
8 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/
9     refs/heads/main/config/mongod.cfg -OutFile "C:\Program Files\MongoDB\Server\8.2\bin\
10    mongod.cfg"
11 Restart-Service MongoDB
12 New-NetFirewallRule -DisplayName "Allow MongoDB 27017" ` 
    -Direction Inbound ` 
    -Protocol TCP ` 
    -LocalPort 27017 ` 
    -Action Allow

```

9.3 Usage

- MongoDB
 - mongosh <public ip of mongodb server>

- SQL Server
 - /opt/mssql-tools18/bin/sqlcmd -S <public ip of sql server> -C -U sa -P mypassword@2025
- PostgreSQL
 - psql -h <public ip of postgresql server> -U postgres
 - password: test

```
3b4840fd2100:/# psql -h 98.92.51.253 -U postgres
Password for user postgres:
psql (17.6, server 11.22)
Type "help" for help.

postgres=# CREATE TABLE person(id INT, name VARCHAR(100));
CREATE TABLE
postgres=# INSERT INTO person VALUES(1, 'james');
INSERT 0 1
postgres=# SELECT * FROM person;
 id | name
----+-----
  1 | james
(1 row)
```

Figure 10: Client using postgresql server

```
mssql@5f1c740a77e7:/$ /opt/mssql-tools18/bin/sqlcmd -S 98.92.51.253 -C -U sa -P
mypassword@2025
1> CREATE TABLE person(id int, name varchar(100))
2> GO
1> INSERT INTO person VALUES(1, 'seakleng')
2> GO

(1 rows affected)
1> INSERT INTO person VALUES(2, 'vanthorn')
2> GO

(1 rows affected)
1> SELECT * FROM person
```

Figure 11: client using sql server

```
root@c482ae26c7e5:/# mongosh 98.92.51.253
Current Mongosh Log ID: 693bdfc3a1e937f09bd861df
Connecting to:          mongodb://98.92.51.253:27017/?directConnection=true&appName=mongosh+2.5.0
Using MongoDB:          8.2.2
Using Mongosh:        2.5.0

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/
```

Figure 12: client using mongodb server

10 Backup Server

10.1 AWS Configuration

Instance Type: t3.medium
Security Group Ports: tcp 3389 (RDP)
Storage: Large EBS volumes or S3 integration
IAM Role: Permissions for S3, EBS snapshots

10.2 Implementation

```
1 Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::  
2   SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex  
3   ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/  
4   install.ps1'))  
5 choco install awscli -y  
6  
7 $bucket = (New-Guid).ToString()  
8 echo $bucket > "C:\bucket.txt"  
9  
10 aws s3 mb "s3://$bucket"  
11  
12 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/  
13   refs/heads/main/scripts/S3Backup.ps1 -OutFile "C:\S3Backup.ps1"  
14 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/  
15   refs/heads/main/scripts/S3BackupSchedule.ps1 -OutFile "C:\S3BackupSchedule.ps1"  
16 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/  
17   refs/heads/main/scripts/S3BackupScheduleNightly.ps1 -OutFile "C:\  
     S3BackupScheduleNightly.ps1"  
18 & "C:\S3BackupSchedule.ps1"  
19 & "C:\S3BackupScheduleNightly.ps1"
```

10.3 Usage

- automatically backup "C:\inetpub" to s3 bucket every 1 minute and every night at 2 a.m.
- check out the s3 bucket

The screenshot shows the Amazon S3 console interface. At the top, there's a navigation bar with 'Amazon S3' and 'Buckets'. Below it, the path '8c400441-ef50-477c-a586-733feceee1d5c' is displayed. On the left, a sidebar shows 'Objects (8)'. The main area is a table titled 'Objects (8)' with columns: Name, Type, Last modified, Size, and Storage class. The table lists several folders and one file:

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	custerr/	Folder	-	-	-
<input type="checkbox"/>	DeviceHealthAttestation/	Folder	-	-	-
<input type="checkbox"/>	ftproot/	Folder	-	-	-
<input type="checkbox"/>	hello.txt	txt	December 12, 2025, 17:29:12 (UTC+07:00)	11.0 B	Standard
<input type="checkbox"/>	history/	Folder	-	-	-
<input type="checkbox"/>	logs/	Folder	-	-	-
<input type="checkbox"/>	temp/	Folder	-	-	-
<input type="checkbox"/>	wwwroot/	Folder	-	-	-

Figure 13: automatically backup to s3 bucket

11 Load Balancing

11.1 AWS Configuration

Service: Application Load Balancer (ALB)
Target Group: Multiple Windows Server instances
Instance Type: t3.medium
Security Group Ports: tcp 3389 (RDP)

11.2 Implementation Steps

11.2.1 Server 1

```
1 Start-Transcript -Path "C:\WebServer.log" -Append
2 Install-WindowsFeature -Name Web-Server
3 # echo "Server 1" > C:\inetpub\wwwroot\index.html
```

11.2.2 Server 2

```
1 Install-WindowsFeature -Name Web-Server -IncludeManagementTools
2 # Install-WindowsFeature -Name Web-Asp-Net45, Web-Net-Ext45
3 # Install-WindowsFeature -Name Web-Mgmt-Console
4 Invoke-WebRequest https://raw.githubusercontent.com/james5635/windows_server_assignment/
   refs/heads/main/static/devspeed.html -OutFile "C:\inetpub\wwwroot\index.html"
```

11.3 Usage

- manually configure load balancer (application load balancer)
- name: devspeedlb
- select all availability Zones and subnets
- create target group (name: web)
 - include WebServerMailServer & Docker
- choose 'web' as target group
- add tcp (port 80) to inbound rule of the load balancer's security group

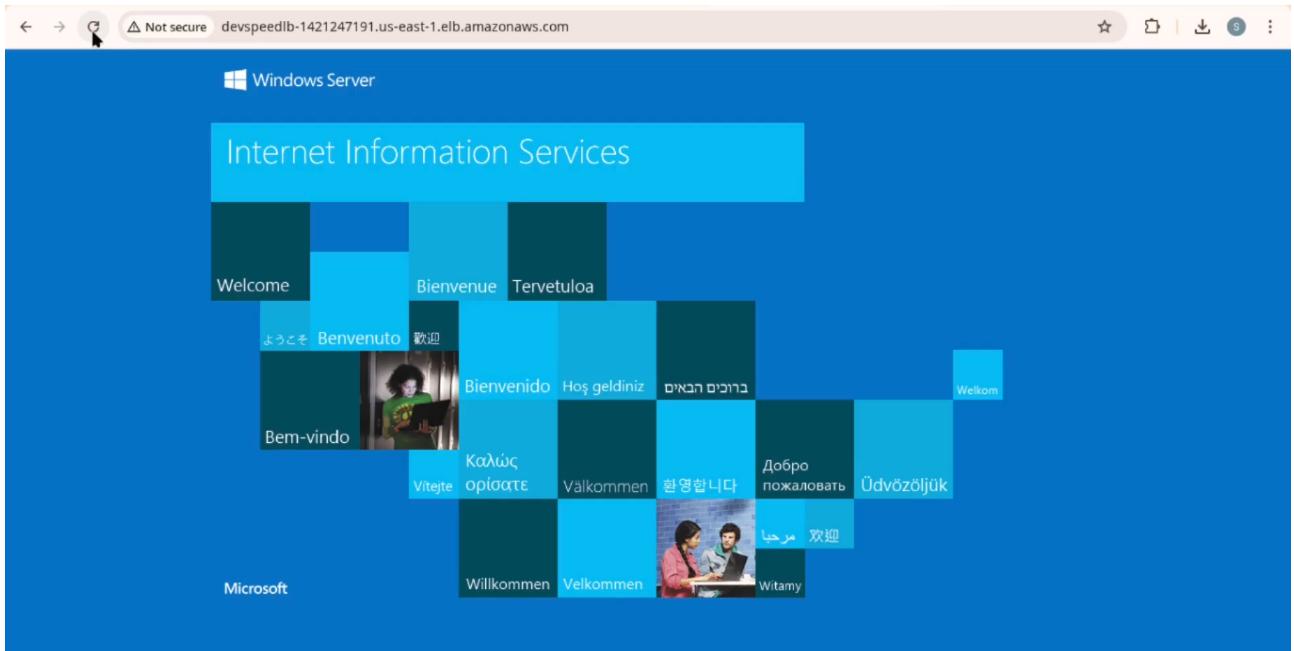


Figure 14: load balance to server 1

A screenshot of the DevSpeed Technologies website. The top navigation bar includes links for "Services", "Infrastructure", "Expertise", and a "Get a Quote" button. The main headline is "Speed Meets Quality" in large blue letters. Below the headline is a subtext: "We deliver high-performance digital solutions—from custom enterprise systems to cloud-native applications—with precision, innovation, and an agile approach." A green call-to-action button at the bottom says "Start Your High-Performance Project >_".

Figure 15: load balance to server 2

12 Failover Cluster

12.1 AWS Configuration

Instance Type: r5.xlarge or larger

Storage: Shared storage using FSx for Windows or S3

Network: Placement groups for low latency

Security Group: Allow cluster communication ports

12.2 Implementation Steps

1. Launch Multiple Windows Server Instances

- Deploy in same VPC, different availability zones
- Use placement group for low latency

2. Install Failover Clustering Feature

```
1 Install-WindowsFeature -Name Failover-Clustering -IncludeManagementTools
```

3. Configure Shared Storage

- **Option A: Amazon FSx for Windows File Server**

Create FSx file system; Mount on all cluster nodes.

- **Option B: EBS Multi-Attach (io2 only)**

Attach same EBS volume to multiple instances; Initialize as cluster shared volume.

4. Create Failover Cluster

```
1 # Validate cluster configuration
2 Test-Cluster -Node "Node1", "Node2"
3
4 # Create cluster
5 New-Cluster -Name "MyCluster" -Node "Node1", "Node2" -StaticAddress "10.0.1.100" -
   NoStorage
```

5. Configure Cluster Quorum

```
1 Set-ClusterQuorum -NodeAndFileShareMajority "\\\FSx\\Witness"
```

6. Add Clustered Role

```
1 # For SQL Server
2 Add-ClusterServerRole -Name "SQL-Cluster" -Storage "Cluster Disk 1"
```

7. Configure Secondary Private IP

- Assign secondary private IP to ENI
- Configure in cluster as virtual IP

12.3 Common Cluster Types in AWS

SQL Server Failover Cluster Use FSx for shared storage; Configure SQL Server on cluster nodes;
Set up availability group for database replication.

File Server Cluster Use FSx or S3 for storage; Configure highly available file shares.

12.4 Best Practices

- Use Amazon FSx for Windows File Server for shared storage
- Deploy cluster nodes in different availability zones
- Use Elastic IP or Network Load Balancer for client access
- Monitor cluster health with CloudWatch
- Regular testing of failover scenarios
- Consider Amazon RDS Multi-AZ for database clustering

13 FTP Server

13.1 AWS Configuration

Instance Name: FTP Server

Instance Type: t3.small to t3.medium

Security Group Ports: tcp 21, tcp 50000-51000, tcp 3389 (RDP)

13.2 Implementation

```
1 #Install IIS Feature
2 Install-WindowsFeature -Name Web-Server -IncludeManagementTools
3
4 #Install FTP feature
5 Install-WindowsFeature -Name Web-Ftp-Server -IncludeAllSubFeature -IncludeManagementTools
   -Verbose
6
7 #Creating new FTP site
8 $SiteName = "Demo FTP Site"
9 $RootFolderPath = "C:\inetpub\ftproot"
10 $PortNumber = 21
11 $FTPUserGroupName = "Demo FTP Users Group"
12 $FTPUserName = "jame"
13 $FTPPassword = ConvertTo-SecureString "Mypassword@2025" -AsPlainText -Force
14
15 if (!(Test-Path $RootFolderPath)) {
16   # if the folder doesn't exist
17   New-Item -Path $RootFolderPath -ItemType Directory # create the folder
18 }
19
20 New-WebFtpSite -Name $SiteName -PhysicalPath $RootFolderPath -Port $PortNumber -Verbose -
  Force
21
22 #Creating the local Windows group
23 if (!(Get-LocalGroup $FTPUserGroupName -ErrorAction SilentlyContinue)) {
24   #if the group doesn't exist
25   New-LocalGroup -Name $FTPUserGroupName ` 
      -Description "Members of this group can connect to FTP server" #create the group
26 }
27
28 # Creating an FTP user
29 If (!(Get-LocalUser $FTPUserName -ErrorAction SilentlyContinue)) {
30   New-LocalUser -Name $FTPUserName -Password $FTPPassword ` 
      -Description "User account to access FTP server" ` 
      -UserMayNotChangePassword
31 }
32
33 # Add the created FTP user to the group Demo FTP Users Group
34 Add-LocalGroupMember -Name $FTPUserGroupName -Member $FTPUserName -ErrorAction
  SilentlyContinue
35
36 # Enabling basic authentication on the FTP site
37 $param = @{
38   Path      = 'IIS:\Sites\Demo FTP Site'
39   Name      = 'ftpserver.security.authentication.basicauthentication.enabled'
40   Value     = $true
41   Verbose   = $True
42 }
43 Set-ItemProperty @param
44
45
46
47
```

```

48 # Adding authorization rule to allow FTP users
49 # in the FTP group to access the FTP site
50 $param = @{
51     PSPPath    = 'IIS:\'
52     Location   = $SiteName
53     Filter     = '/system.ftpserver/security/authorization'
54     # Value     = @{} accesstype = 'Allow'; roles = $FTPUserGroupName; permissions = 1 }
55     Value      = @{} accesstype = 'Allow'; roles = $FTPUserGroupName; permissions = 3 }
56 }
57
58 Add-WebConfiguration @param
59
60 # Changing SSL policy of the FTP site
61 'ftpServer.security.ssl.controlChannelPolicy', 'ftpServer.security.ssl.dataChannelPolicy'
62 |
63 ForEach-Object {
64     Set-ItemProperty -Path "IIS:\Sites\Demo FTP Site" -Name $_ -Value $false
65 }
66
67 # can be 'ReadAndExecute', 'Modify'
68 $ACLObject = Get-Acl -Path $RootFolderPath
69 $ACLObject.SetAccessRule(
70     ( # Access rule object
71         New-Object System.Security.AccessControl.FileSystemAccessRule(
72             $FTPUserGroupName,
73             'FullControl',
74             'ContainerInherit, ObjectInherit',
75             'None',
76             'Allow'
77         )
78     )
79 )
80 Set-Acl -Path $RootFolderPath -AclObject $ACLObject
81
82 # Checking the NTFS permissions on the FTP root folder
83 Get-Acl -Path $RootFolderPath | ForEach-Object Access
84
85 # Test FTP Port and FTP access
86 Test-NetConnection -ComputerName localhost -Port 21
87
88 Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.ftpServer
89 /firewallSupport" -name "lowDataChannelPort" -value 50000
90 Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.ftpServer
91 /firewallSupport" -name "highDataChannelPort" -value 51000
92
93 Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.
94 applicationHost/sites/site[@name='Demo FTP Site']/ftpServer/firewallSupport" -name "externalIpAddress" -value "${MyEIP}"
95
96 # Control port
97 New-NetFirewallRule -DisplayName "Allow FTP 21" -Direction Inbound -Protocol TCP -
98     LocalPort 21 -Action Allow
99
100 # Passive ports
101 New-NetFirewallRule -DisplayName "Allow FTP Passive Ports" -Direction Inbound -Protocol
102     TCP -LocalPort 50000-51000 -Action Allow
103
104 Restart-Service FTPSVC
105
106 # C:\Windows\System32\config\systemprofile\AppData\Local\Temp\EC2Launch380802110\
107     UserScript.ps1
108 # apple

```

```
103  
104 echo hello > "C:\inetpub\ftproot\hello.txt"
```

13.3 Usage

- use filezilla (port 21)
- username: jame
- password: Mypassword@2025

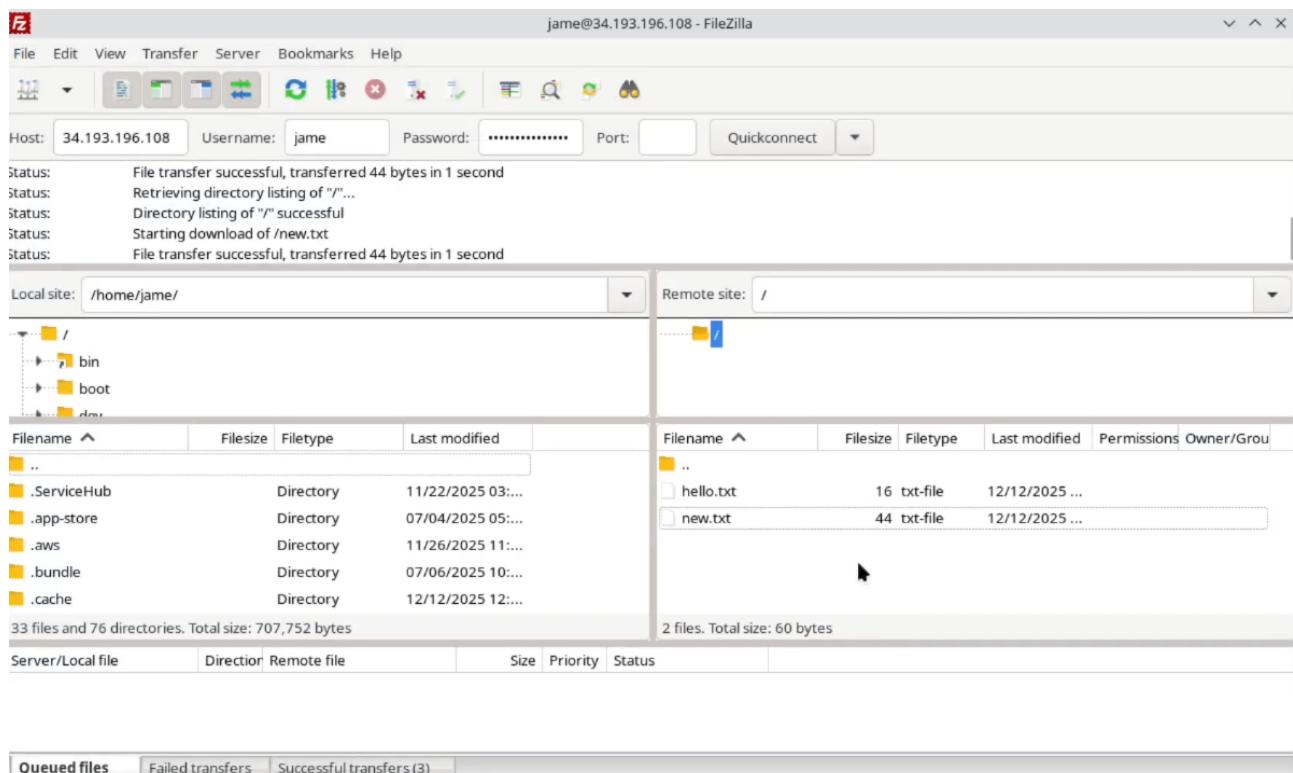


Figure 16: FileZilla successfully connected to ftp server

14 Container (Docker)

14.1 AWS Configuration

Instance Type: t3.medium or larger

Operating System: Windows Server 2019/2022 with Containers

Security Group Ports: Custom ports based on containerized applications

14.2 Implementation

```
1 Invoke-WebRequest -UseBasicParsing "https://raw.githubusercontent.com/microsoft/Windows-Containers/Main/helpful_tools/Install-DockerCE/install-docker-ce.ps1" -o install-docker-ce.ps1
2 .\install-docker-ce.ps1
3 # will restart the machine
```

14.3 Usage

- connect to the Docker ec2 instance
- docker pull mcr.microsoft.com/windows/nanoserver:ltsc2022
- docker pull mcr.microsoft.com/windows/servercore:ltsc2022 (optional)
- docker run -it <image>

The screenshot shows a Windows PowerShell window with the following command history:

```
Administrator: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
PS C:\Windows\system32> docker pull mcr.microsoft.com/windows/nanoserver:ltsc2022
ltsc2022: Pulling from windows/nanoserver
c7c5ba939832: Pull complete
Digest: sha256:643adf84ee2338ee4811fd891adb9e912917dc6c0ca85399982e1bebda4f2295
Status: Downloaded newer image for mcr.microsoft.com/windows/nanoserver:ltsc2022
mcr.microsoft.com/windows/nanoserver:ltsc2022
PS C:\Windows\system32> docker images


| IMAGE                                         | ID           | DISK USAGE | CONTENT SIZE | EXTRA |
|-----------------------------------------------|--------------|------------|--------------|-------|
| mcr.microsoft.com/windows/nanoserver:ltsc2022 | 704edc10ed58 | 305MB      | 0B           |       |


PS C:\Windows\system32> docker run -it mcr.microsoft.com/windows/nanoserver:ltsc2022
```

Figure 17: Running windows server container

15 Domain Controller

15.1 AWS Configuration

Instance Name: Domain Controller

Instance Type: t3.medium or larger

Security Group Ports:

- tcp 80
- tcp 3389 (RDP)
- tcp 0-65535 (All tcp ports)
- udp 0-65535 (all udp ports)
- icmp -1 (all icmp)

Storage: Minimum 50GB SSD

Operating System: Windows Server 2019/2022

15.2 Implementation

```
1 Install-WindowsFeature -Name AD-Domain-Services -IncludeManagementTools
2 Install-ADDSForest -DomainName "example.local" -InstallDNS -SafeModeAdministratorPassword
   (ConvertTo-SecureString "Mypassword@2025" -AsPlainText -Force) -NoRebootOnCompletion
   -Force
3 # Get-Service adws,kdc,netlogon,dns
4 Restart-Computer
```

15.3 Usage

- connect to File Server or other windows server
- change dns server address to <private ip of domain controller>
- join the domain 'example.local'

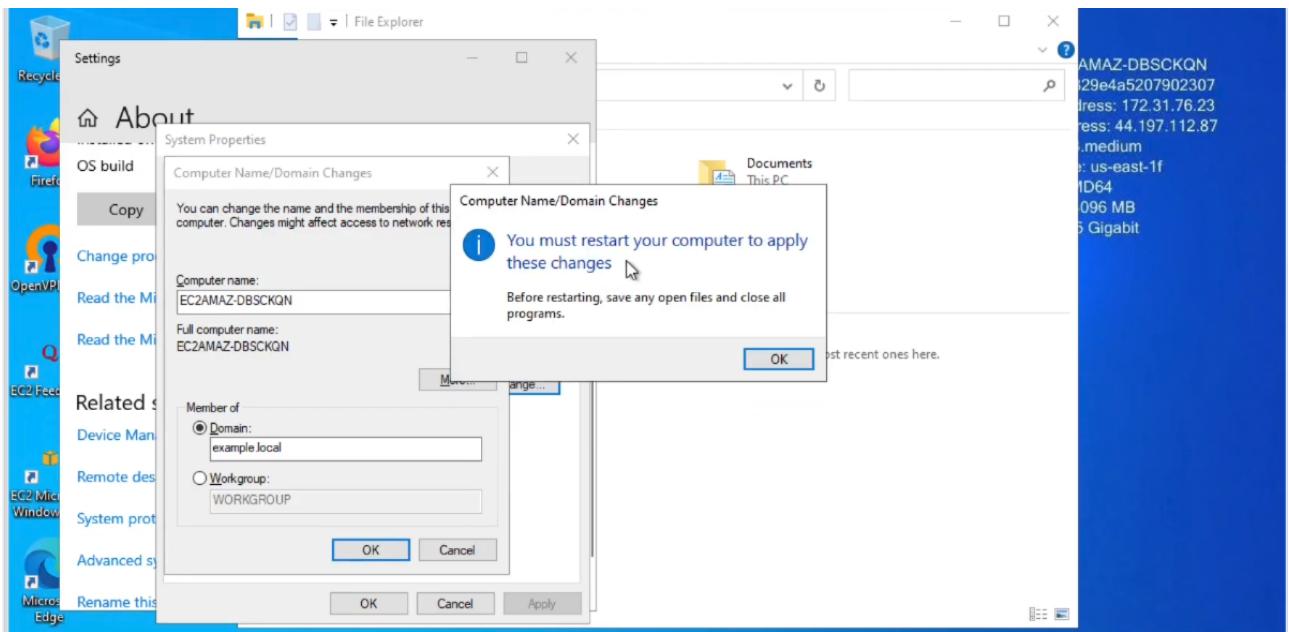


Figure 18: Successfully join the domain

16 DevSpeedLLM

16.1 Configuration

GPU: dual NVIDIA T4 GPUs

OS: Linux

16.2 Implementation

```
1 from datasets import load_dataset
2
3 # train_dataset = load_dataset("trl-lib/Capybara", split="train")
4 def make_chatml(example):
5     return {
6         "messages": [ [ {"role": "user", "content": "Summarize the following text:\n\n" +
7             content },
8                         {"role": "assistant", "content": summary } ] +
9                         [x for qa in qas for x in (
10                             {"role": "user", "content": qa["question"] if qa['question'] !=
11                             None else ""}, # There are 2 question=None in the dataset
12                             {"role": "assistant", "content": qa["answer"]}),
13                         ] for content, summary, qas in zip(example['content'],
14                         example['summary'], example["QAs"])
15                     ]
16     }
17
18 dataset = load_dataset("james56352025/devspeedllm-datasets", "rupp_enhanced", split = "train")
19 train_dataset = dataset.map(make_chatml, batched=True, remove_columns=['summary', 'qa_pairs', 'content', 'QAs'])
20
21
22 from transformers import AutoModelForCausalLM, BitsAndBytesConfig
23 import torch
24
25 # model_id = "Qwen/Qwen2.5-1.5B-Instruct"
26 # model_id = "mistralai/Mistral-7B-Instruct-v0.1"
27 # model_id = "google/gemma-3-270m-it"
28 model_id = "meta-llama/Llama-3.2-3B-Instruct"
29 output_dir = "lora-out"
30
31 four_bit = BitsAndBytesConfig(
32     load_in_4bit=True,                                     # Load the model in 4-bit precision to
33     save_memory,                                         # Data type used for internal
34     bnb_4bit_compute_dtype=torch.float16,                 # computations in quantization
35     bnb_4bit_use_double_quant=True,                       # Use double quantization to improve
36     accuracy,                                            # Type of quantization. "nf4" is
37     bnb_4bit_quant_type="nf4"                            # recommended for recent LLMs
38 )
39
40 model = AutoModelForCausalLM.from_pretrained(
41     model_id,                                              # Change to Flash Attention if GPU has
42     attnImplementation="sdpa",                            # support
43     dtype=torch.float16,                                  # Change to bfloat16 if GPU has support
44     useCache=True,                                       # Whether to cache attention outputs to
45     speedUpInference=True,                                # quantization_config=four_bit,
46 )
```

```

42 from peft import LoraConfig
43
44 # You may need to update `target_modules` depending on the architecture of your chosen
45 # model.
46 # For example, different LLMs might have different attention/projection layer names.
47 peft_config = LoraConfig(
48     r=32,
49     lora_alpha=32,
50     target_modules = ["q_proj", "k_proj", "v_proj", "o_proj", "gate_proj", "up_proj", "down_proj",],
51 )
52
53 import shutil, os, glob
54 from transformers.trainer_callback import TrainerCallback
55
56 class ZipCallback(TrainerCallback):
57     # Required empty methods
58     def on_init_end(self, args, state, control, **kwargs):
59         pass
60     def on_train_begin(self, args, state, control, **kwargs):
61         pass
62     def on_train_end(self, args, state, control, **kwargs):
63         pass
64     def on_epoch_end(self, args, state, control, **kwargs):
65         pass
66     # THE ONE WE USE
67     def on_save(self, args, state, control, **kwargs):
68         # Find latest checkpoint
69         # ckpts = sorted(glob.glob(os.path.join(args.output_dir, "checkpoint-*")))
70         ckpts = glob.glob(os.path.join(args.output_dir, "checkpoint-*"))
71         if not ckpts:
72             return control
73         # latest = ckpts[-1]
74         latest = max(ckpts, key=lambda x: int(x.split("-")[-1]))
75         zip_path = os.path.join(os.path.dirname(args.output_dir), os.path.basename(latest)
76 ) + ".zip")
77         if os.path.exists(zip_path):
78             print("\n" + f"Already zipped: {zip_path}")
79             return control
80         shutil.make_archive(os.path.splitext(zip_path)[0], "zip", latest)
81         print()
82         print(f"Zipped: {zip_path}")
83         return control
84
85
86 from trl import SFTConfig, SFTTrainer
87
88 training_args = SFTConfig(
89     # Training schedule / optimization
90     per_device_train_batch_size = 1,           # Batch size per GPU
91     gradient_accumulation_steps = 4,          # Gradients are accumulated over multiple steps
92     # → effective batch size = 2 * 8 = 16
93     warmup_steps = 5,                         # Number of full dataset passes. For shorter
94     num_train_epochs = 5,                      # training, use `max_steps` instead (this case)
95     # max_steps = 2,
96     learning_rate = 2e-4,                     # Learning rate for the optimizer
97     optim = "paged_adamw_8bit",               # Optimizer
98     # Logging / reporting
99     logging_steps=1,                          # Log training metrics every N steps
100    report_to="trackio",                     # Experiment tracking tool
101    trackio_space_id=output_dir,              # HF Space where the experiment tracking will
102    be saved

```

```

99     output_dir=output_dir,
100    # max_length=1024,
101    use_liger_kernel=True,
102    training
103    activation_offloading=True,
104    memory_usage
105    gradient_checkpointing=True,
106    during_backpropagation
107    # Hub integration
108    push_to_hub=False,
109    Hugging Face Hub
110    # The model will be saved under your Hub
111    account_in_the_repository_named `output_dir`
112    gradient_checkpointing_kwargs={"use_reentrant": False}, # To prevent warning message
113 )
114
115 trainer = SFTTrainer(
116     model=model,
117     args=training_args,
118     train_dataset=train_dataset,
119     peft_config=peft_config,
120     callbacks = [ZipCallback()],
121 )
122
123 trainer_stats = trainer.train()
124 trainer.save_model(output_dir)

```

16.3 Usage

- ask "tell me about RUPP"
- ask "list academic staffs of computer science department at RUPP with name"
- ask "teach me flutter and give example"

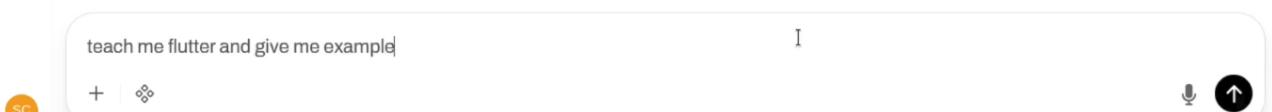
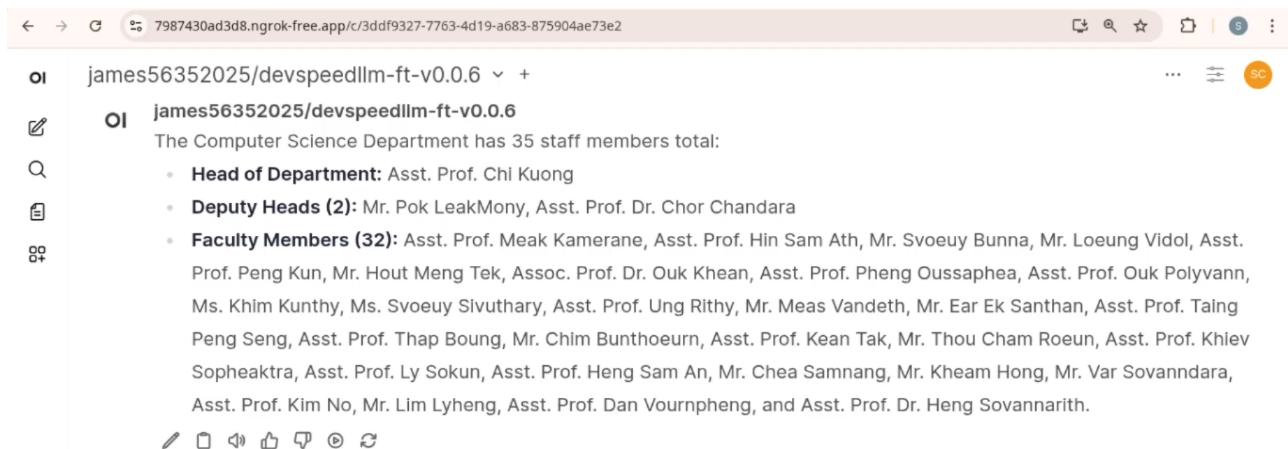


Figure 19: Chatting with DevSpeedLLM