Launch and Manage an Amazon EC2 Instance Using AWS CLI

1. Introduction

In this lab assignment, I explored Amazon EC2 (Elastic Compute Cloud) using AWS Command Line Interface. EC2 allows us to create and manage virtual servers in the cloud. Instead of using the AWS web console, I used CLI commands which are more suitable for automation and real-world DevOps scenarios.2. What This Lab Covers

I learned how to:

- Configure AWS CLI securely
- Verify AWS identity access
- Create key pairs and security groups
- Find Free Tier–eligible AMIs
- Launch an EC2 instance using CLI
- Connect to EC2 via SSH
- Monitor Free Tier usage
- Clean up resources to avoid billing

3. Why Use AWS CLI Instead of Console?

Reason	**Explanation**
Automation	Repeat deployments using scripts
Speed	Faster than clicking through UI
DevOps Ready	Required for CI/CD pipelines
Accuracy	Less human error in large setups
Industry Standard	Used in real cloud projects

4. Prerequisites

Before starting, ensure:

- Active AWS account with **Free Tier**

- IAM user with:
  - Programmatic access
  - AmazonEC2FullAccess
- Local machine with:
  - Linux / macOS / Windows (WSL recommended)
  - Internet access

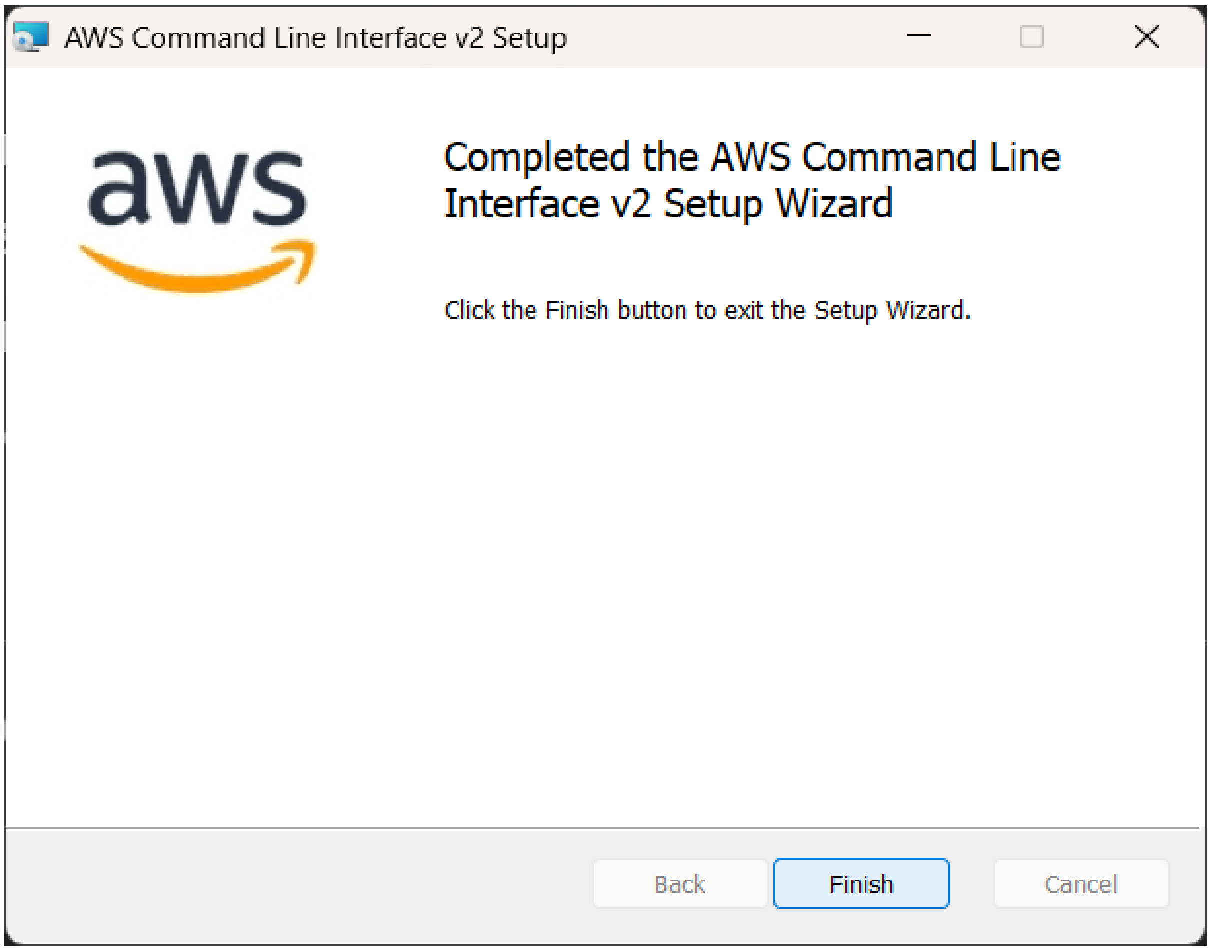
5. Step 1 – Prepare the Environment

5.1 Install AWS CLI

Install AWS CLI v2 from the official AWS documentation (OS-specific).

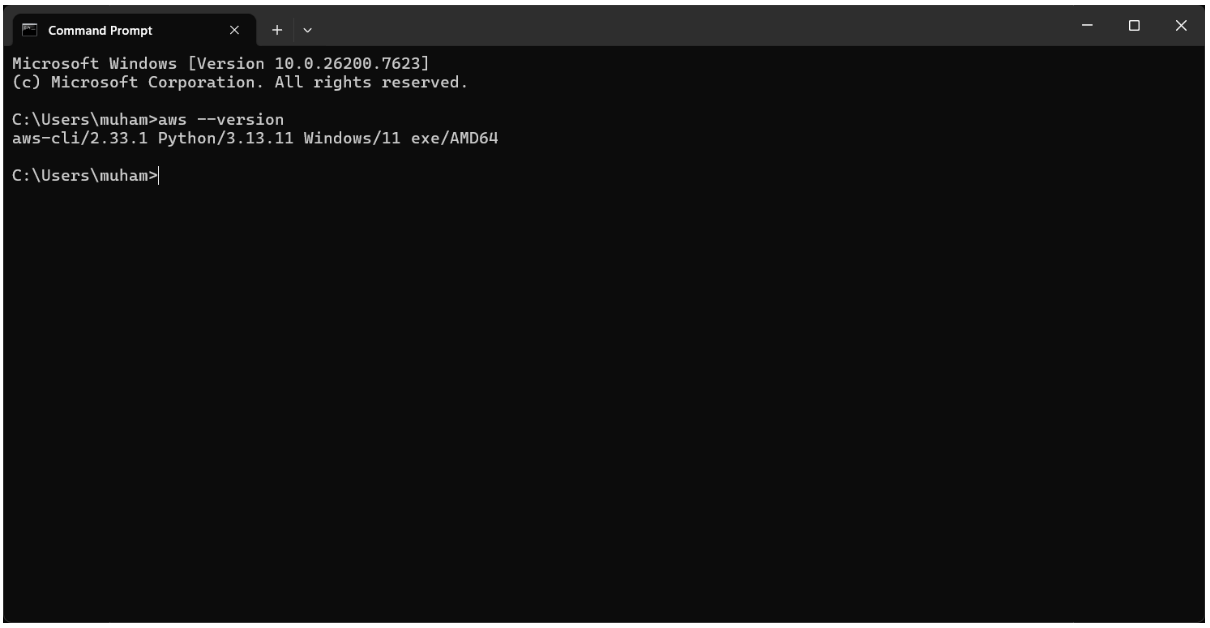
Verify installation:

```
aws --version
```



Completed the AWS Command Line Interface v2 Setup Wizard

Click the Finish button to exit the Setup Wizard.



```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\muham>aws --version
aws-cli/2.33.1 Python/3.13.11 Windows/11 exe/AMD64

C:\Users\muham>
```

5.2 Configure AWS CLI

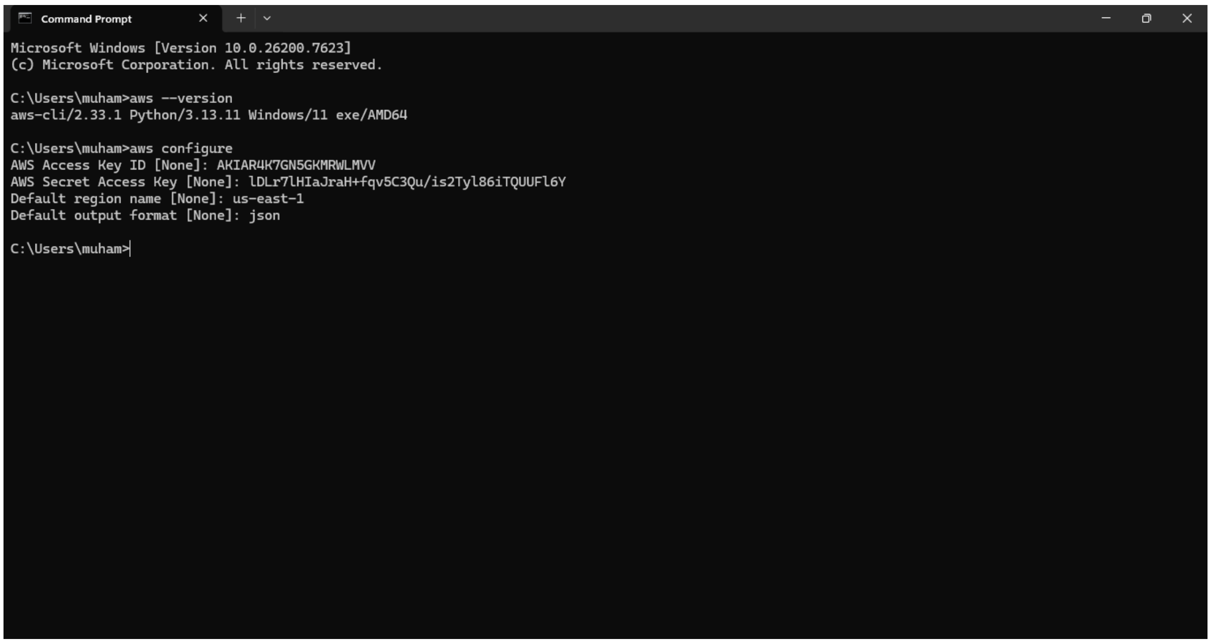
Run:

```
aws configure
```

Provide:

- **AWS Access Key ID**
- **AWS Secret Access Key**
- **Default region** (example: us-east-1)
- **Output format**: json

This securely stores credentials in ~/.aws/credentials.



```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\muham>aws --version
aws-cli/2.33.1 Python/3.13.11 Windows/11 exe/AMD64

C:\Users\muham>aws configure
AWS Access Key ID [None]: AKIAR4K7GN5GKMRWLMVV
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json

C:\Users\muham>
```

6. Step 2 – Verify AWS CLI Access

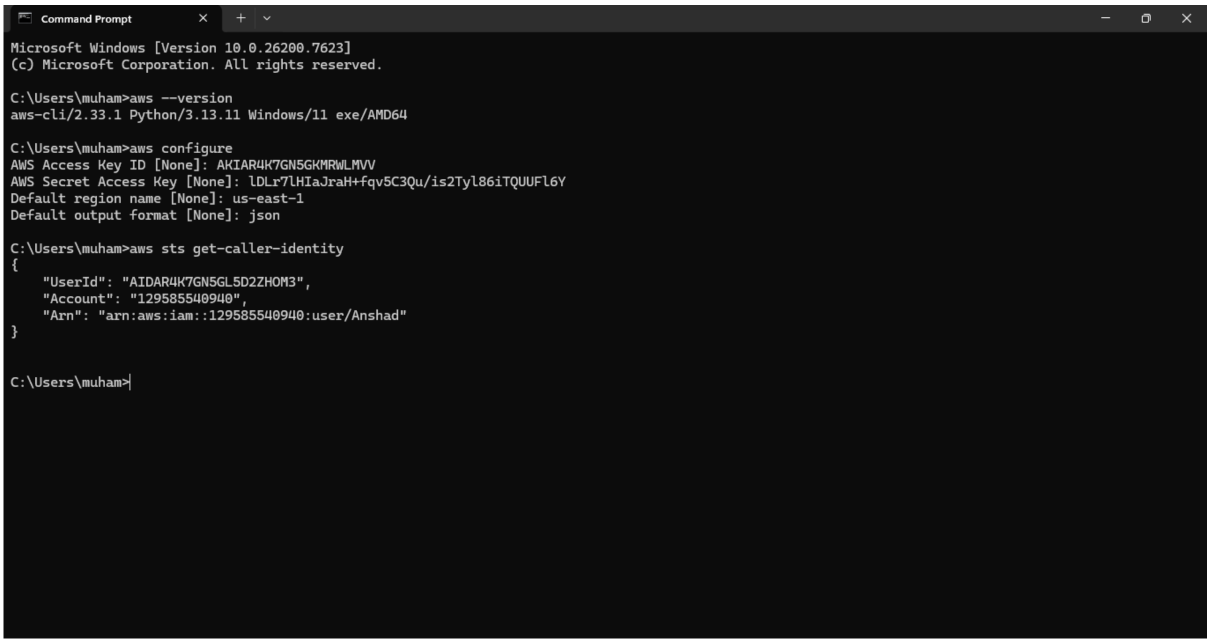
Confirm connectivity and identity:

aws sts get-caller-identity

Expected Output:

- Account ID
- IAM User ARN

Successful output confirms valid credentials and permissions.



```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\muham>aws --version
aws-cli/2.33.1 Python/3.13.11 Windows/11 exe/AMD64

C:\Users\muham>aws configure
AWS Access Key ID [None]: AKIAR4K7GN5GKMRWLMVV
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json

C:\Users\muham>aws sts get-caller-identity
{
    "UserId": "AIDAR4K7GN5GL5D2ZHOM3",
    "Account": "129585540940",
    "Arn": "arn:aws:iam::129585540940:user/Anshad"
}


C:\Users\muham>
```

7. Step 3 – Create a Key Pair (SSH Access)

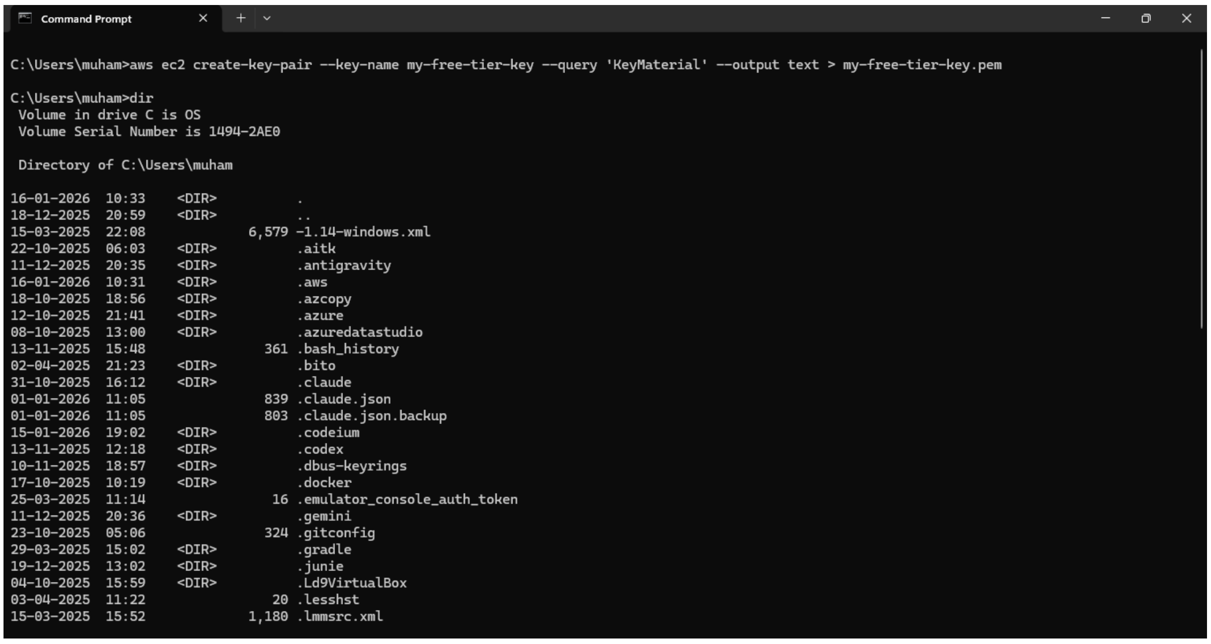
7.1 Create Key Pair

```
aws ec2 create-key-pair \
  --key-name my-free-tier-key \
  --query 'KeyMaterial' \
  --output text > my-free-tier-key.pem
```

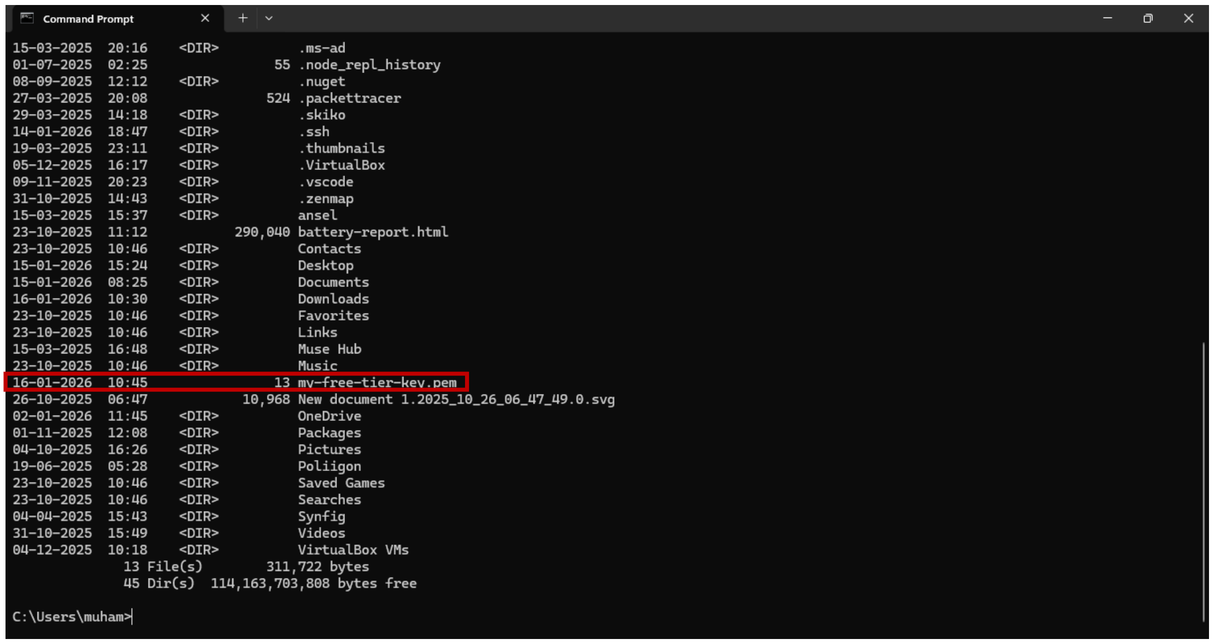
7.2 Secure Key Permissions

```
chmod 400 my-free-tier-key.pem
```

Why this is required:
SSH refuses insecure private key permissions.



```
C:\Users\muham>aws ec2 create-key-pair --key-name my-free-tier-key --query 'KeyMaterial' --output text > my-free-tier-key.pem

C:\Users\muham>dir
 Volume in drive C is OS
 Volume Serial Number is 1494-2AE0

 Directory of C:\Users\muham

16-01-2026  10:33    <DIR>          .
18-12-2025  20:59    <DIR>          ..
15-03-2025  22:08             6,579 -1.14-windows.xml
22-10-2025  06:03    <DIR>          .aitk
11-12-2025  20:35    <DIR>          .antigravity
16-01-2026  10:31    <DIR>          .aws
18-10-2025  18:56    <DIR>          .azcopy
12-10-2025  21:41    <DIR>          .azure
08-10-2025  13:00    <DIR>          .azuredatastudio
13-11-2025  15:48               361 .bash_history
02-04-2025  21:23    <DIR>          .bito
31-10-2025  16:12    <DIR>          .claude
01-01-2026  11:05               839 .claude.json
01-01-2026  11:05               803 .claude.json.backup
15-01-2026  19:02    <DIR>          .codeium
13-11-2025  12:18    <DIR>          .codex
10-11-2025  18:57    <DIR>          .dbus-keyrings
17-10-2025  10:19    <DIR>          .docker
25-03-2025  11:14                16 .emulator_console_auth_token
11-12-2025  20:36    <DIR>          .gemini
23-10-2025  05:06               324 .gitconfig
29-03-2025  15:02    <DIR>          .gradle
19-12-2025  13:02    <DIR>          .junie
04-10-2025  15:59    <DIR>          .Ld9VirtualBox
03-04-2025  11:22                20 .lesshst
15-03-2025  15:52             1,180 .lmmsrc.xml
```



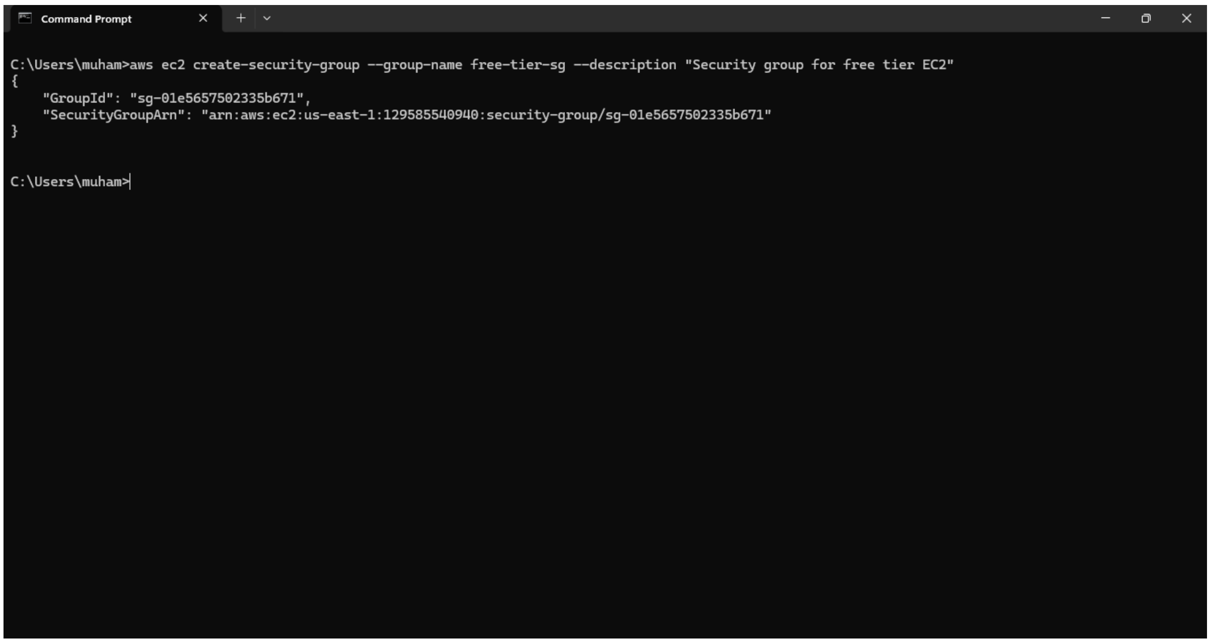
```
15-03-2025  20:16    <DIR>          .ms-ad
01-07-2025  02:25                55 .node_repl_history
08-09-2025  12:12    <DIR>          .nuget
27-03-2025  20:08               524 .packettracer
29-03-2025  14:18    <DIR>          .skiko
14-01-2026  18:47    <DIR>          .ssh
19-03-2025  23:11    <DIR>          .thumbnails
05-12-2025  16:17    <DIR>          .VirtualBox
09-11-2025  20:23    <DIR>          .vscode
31-10-2025  14:43    <DIR>          .zenmap
15-03-2025  15:37    <DIR>          ansel
23-10-2025  11:12           290,040 battery-report.html
23-10-2025  10:46    <DIR>          Contacts
15-01-2026  15:24    <DIR>          Desktop
15-01-2026  08:25    <DIR>          Documents
16-01-2026  10:30    <DIR>          Downloads
23-10-2025  10:46    <DIR>          Favorites
23-10-2025  10:46    <DIR>          Links
15-03-2025  16:48    <DIR>          Muse Hub
23-10-2025  10:46    <DIR>          Music
16-01-2026  10:45                13 my-free-tier-key.pem
26-10-2025  06:47            10,968 New document 1.2025_10_26_06_47_49.0.svg
02-01-2026  11:45    <DIR>          OneDrive
01-11-2025  12:08    <DIR>          Packages
04-10-2025  16:26    <DIR>          Pictures
19-06-2025  05:28    <DIR>          Poliigon
23-10-2025  10:46    <DIR>          Saved Games
23-10-2025  10:46    <DIR>          Searches
04-04-2025  15:43    <DIR>          Synfig
31-10-2025  15:49    <DIR>          Videos
04-12-2025  10:18    <DIR>          VirtualBox VMs
              13 File(s)        311,722 bytes
              45 Dir(s)  114,163,703,808 bytes free

C:\Users\muham>
```

8. Step 4 – Create a Security Group

8.1 Create Security Group

```
aws ec2 create-security-group \
 --group-name free-tier-sg \
 --description "Security group for Free Tier EC2"
```

Security Groups act as **virtual firewalls**.

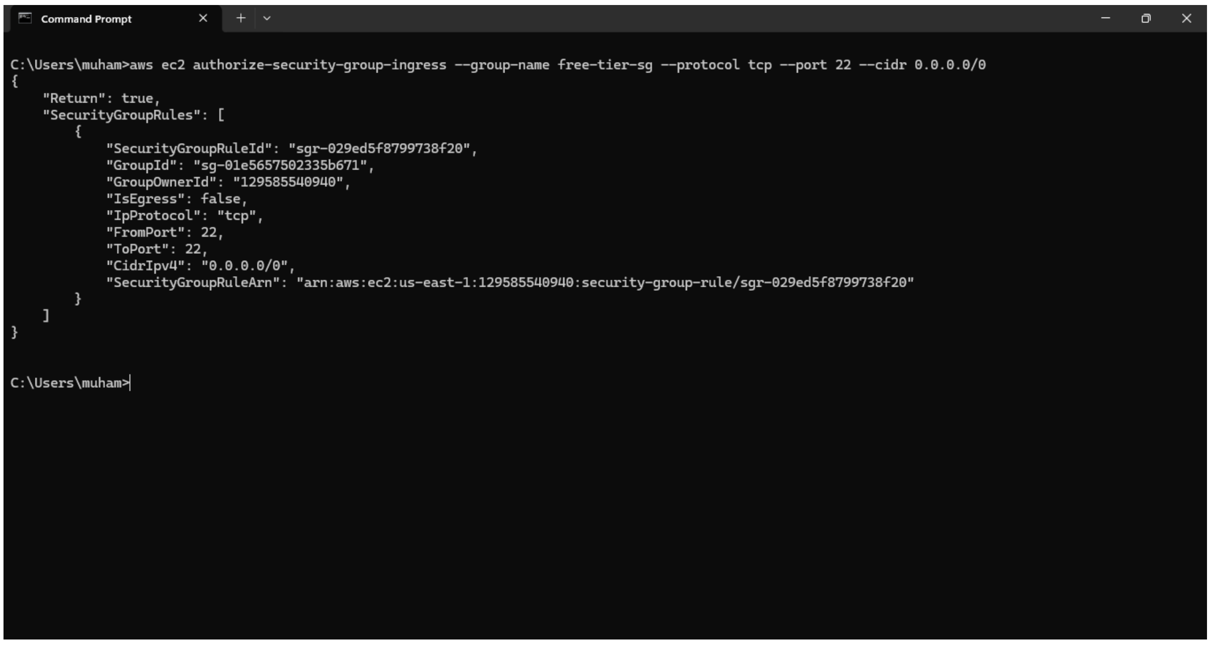


```
C:\Users\muham>aws ec2 create-security-group --group-name free-tier-sg --description "Security group for free tier EC2"
{
    "GroupId": "sg-01e5657502335b671",
    "SecurityGroupArn": "arn:aws:ec2:us-east-1:129585540940:security-group/sg-01e5657502335b671"
}


C:\Users\muham>
```

8.2 Allow SSH Access (Port 22)

```
aws ec2 authorize-security-group-ingress \
 --group-name free-tier-sg \
 --protocol tcp \
 --port 22 \
 --cidr 0.0.0.0/0
```

Open to all IPs for learning purposes only.

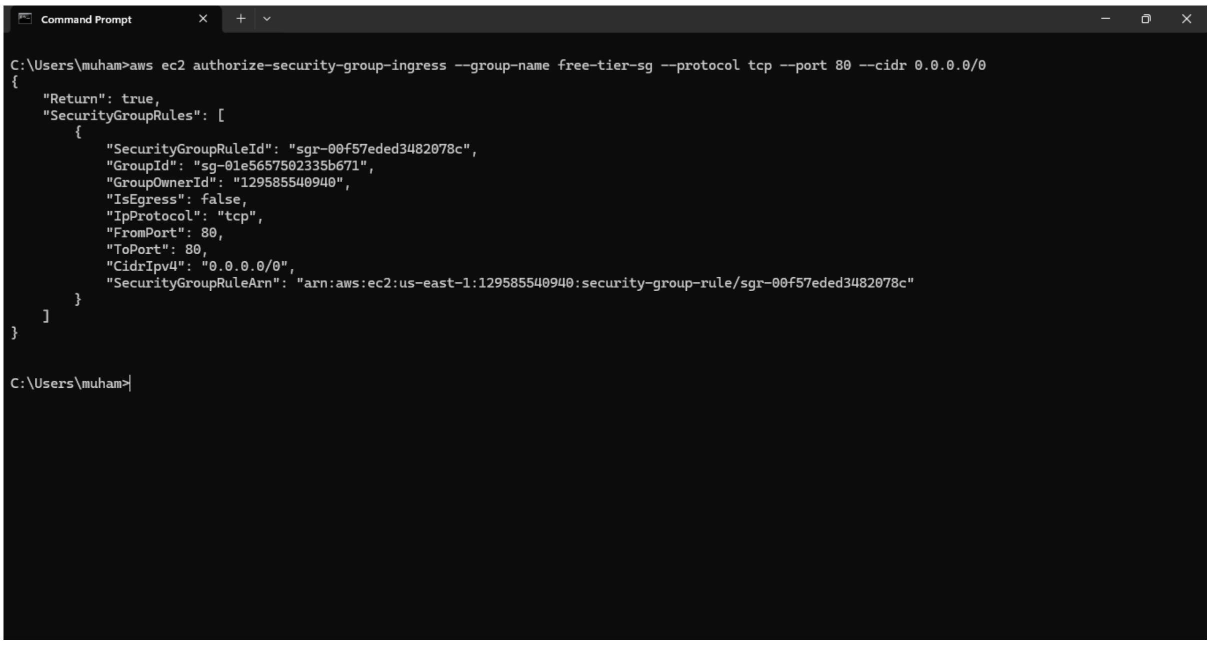


```
C:\Users\muham>aws ec2 authorize-security-group-ingress --group-name free-tier-sg --protocol tcp --port 22 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-029ed5f8799738f20",
            "GroupId": "sg-01e5657502335b671",
            "GroupOwnerId": "129585540940",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "0.0.0.0/0",
            "SecurityGroupRuleArn": "arn:aws:ec2:us-east-1:129585540940:security-group-rule/sgr-029ed5f8799738f20"
        }
    ]
}


C:\Users\muham>
```

8.3 (Optional) Allow HTTP Access

```
aws ec2 authorize-security-group-ingress \
 --group-name free-tier-sg \
 --protocol tcp \
 --port 80 \
 --cidr 0.0.0.0/0
```

Used for web servers.



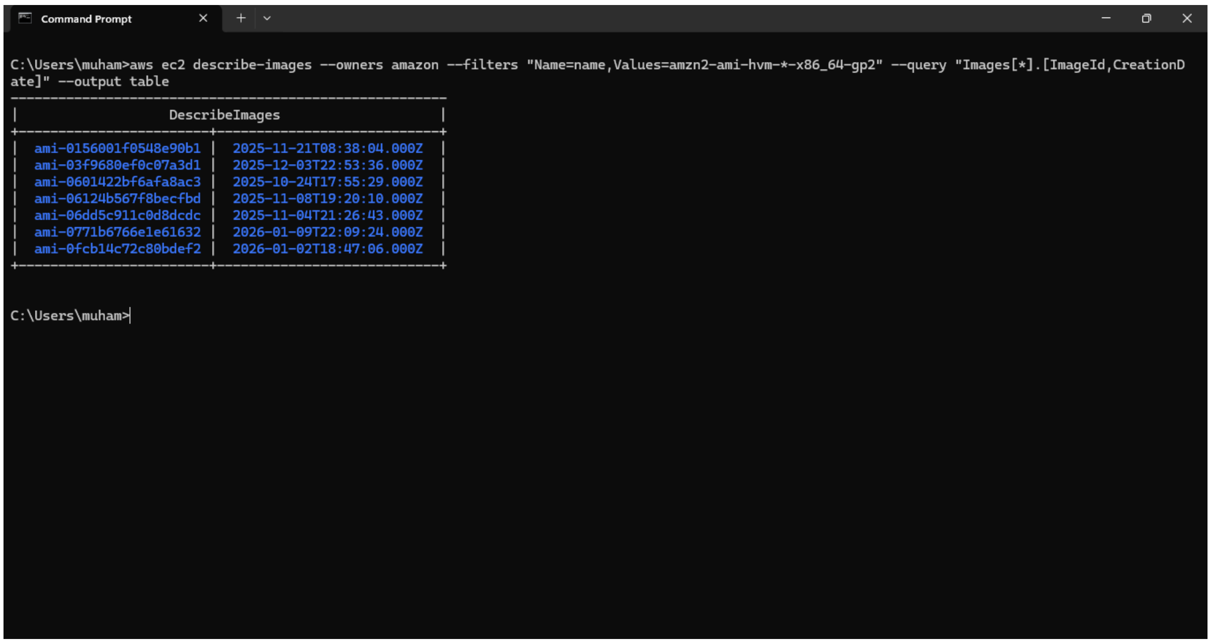
```
C:\Users\muham>aws ec2 authorize-security-group-ingress --group-name free-tier-sg --protocol tcp --port 80 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-00f57eded3482078c",
            "GroupId": "sg-01e5657502335b671",
            "GroupOwnerId": "129585540940",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "0.0.0.0/0",
            "SecurityGroupRuleArn": "arn:aws:ec2:us-east-1:129585540940:security-group-rule/sgr-00f57eded3482078c"
        }
    ]
}


C:\Users\muham>
```

9. Step 5 – Find a Free Tier Eligible AMI

List Amazon Linux 2 AMIs:

```
aws ec2 describe-images \
 --owners amazon \
 --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2" \
 --query 'Images[*].[ImageId,CreationDate]' \
 --output table
```

Note the **latest AMI ID** (newest creation date).



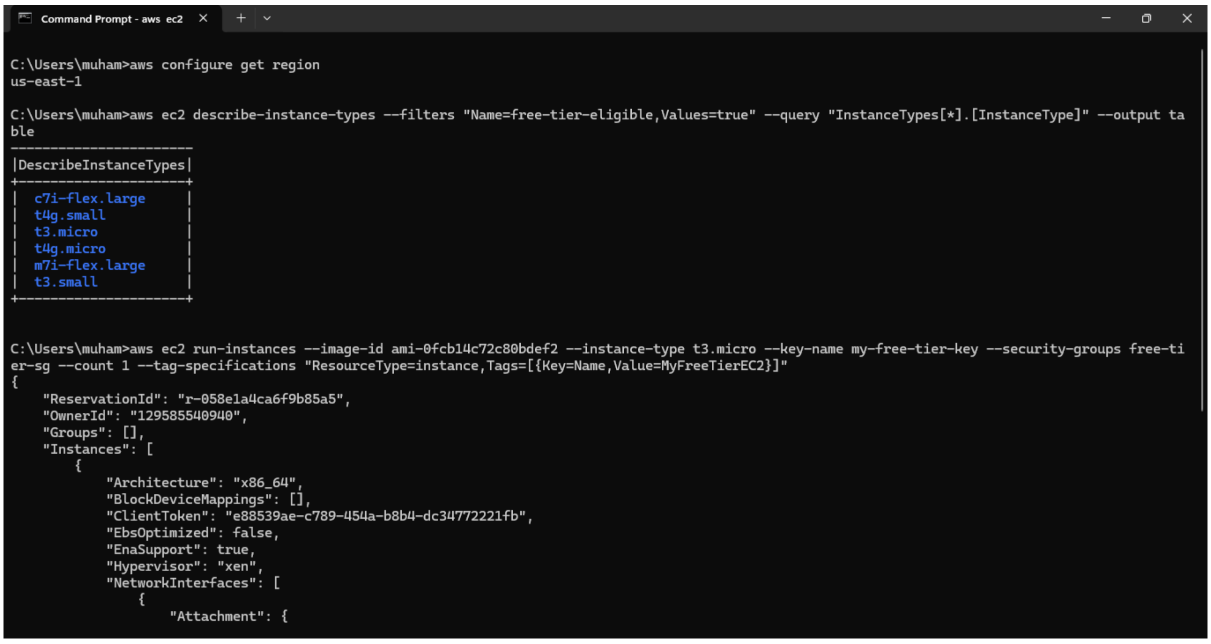
```
C:\Users\muham>aws ec2 describe-images --owners amazon --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2" --query "Images[*].[ImageId,CreationD
ate]" --output table
-------------------------------------------------------
|                   DescribeImages                    |
+------------------------+----------------------------+
|  ami-0156001f0548e90b1 |  2025-11-21T08:38:04.000Z  |
|  ami-03f9680ef0c07a3d1 |  2025-12-03T22:53:36.000Z  |
|  ami-0601422bf6afa8ac3 |  2025-10-24T17:55:29.000Z  |
|  ami-06124b567f8becfbd |  2025-11-08T19:20:10.000Z  |
|  ami-06dd5c911c0d8dcdc |  2025-11-04T21:26:43.000Z  |
|  ami-0771b6766e1e61632 |  2026-01-09T22:09:24.000Z  |
|  ami-0fcb14c72c80bdef2 |  2026-01-02T18:47:06.000Z  |
+------------------------+----------------------------+


C:\Users\muham>
```

10. Step 6 – Launch a Free Tier EC2 Instance

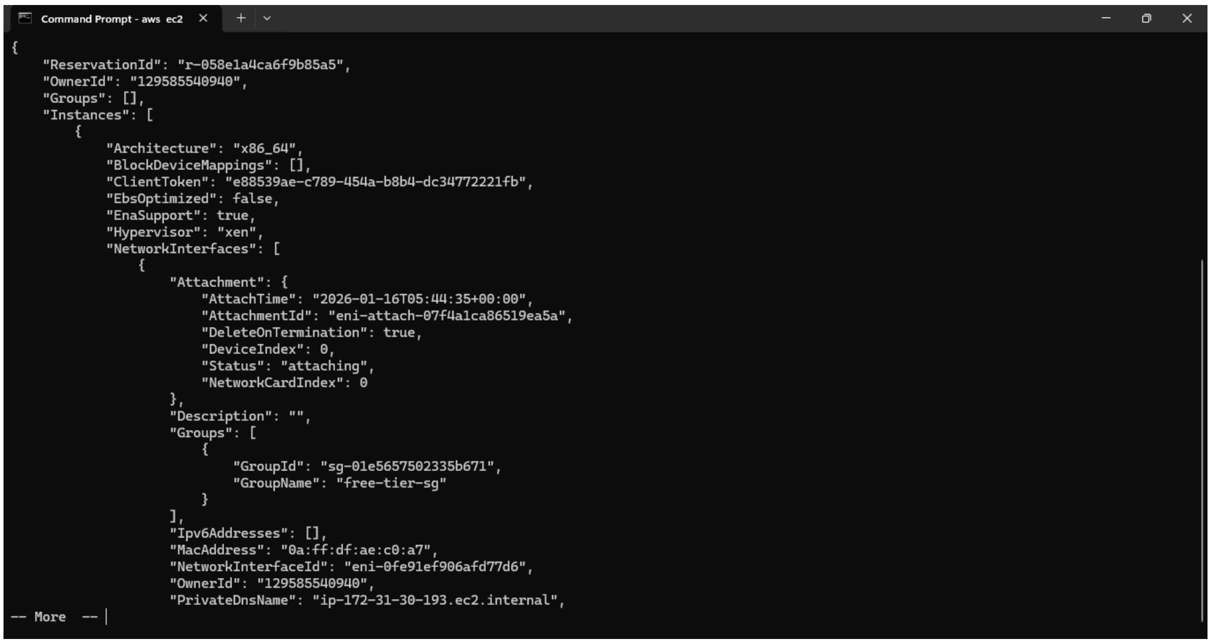
```
aws ec2 run-instances \
 --image-id ami-xxxxxxxx \
 --instance-type t2.micro \
 --key-name my-free-tier-key \
 --security-groups free-tier-sg \
 --count 1 \
 --tag-specifications
'ResourceType=instance,Tags=[{Key=Name,Value=MyFreeTierEC2}]'
```

Why t2.micro?

- Free Tier eligible
- 1 vCPU, 1 GB RAM
- Ideal for learning and testing



```
C:\Users\muham>aws configure get region
us-east-1

C:\Users\muham>aws ec2 describe-instance-types --filters "Name=free-tier-eligible,Values=true" --query "InstanceTypes[*].[InstanceType]" --output ta
ble
-----------------------
|DescribeInstanceTypes|
+---------------------+
|  c7i-flex.large     |
|  t4g.small          |
|  t3.micro           |
|  t4g.micro          |
|  m7i-flex.large     |
|  t3.small           |
+---------------------+


C:\Users\muham>aws ec2 run-instances --image-id ami-0fcb14c72c80bdef2 --instance-type t3.micro --key-name my-free-tier-key --security-groups free-ti
er-sg --count 1 --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=MyFreeTierEC2}]"
{
    "ReservationId": "r-058e1a4ca6f9b85a5",
    "OwnerId": "129585540940",
    "Groups": [],
    "Instances": [
        {
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "e88539ae-c789-454a-b8b4-dc34772221fb",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
```



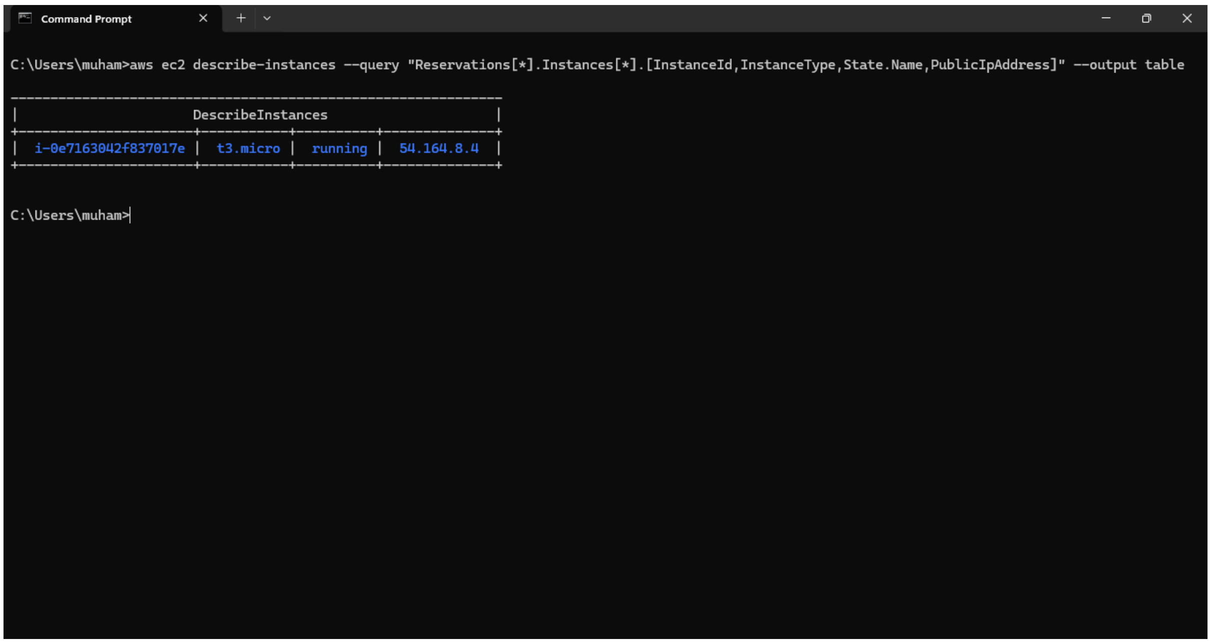
```
{
    "ReservationId": "r-058e1a4ca6f9b85a5",
    "OwnerId": "129585540940",
    "Groups": [],
    "Instances": [
        {
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "e88539ae-c789-454a-b8b4-dc34772221fb",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
                        "AttachTime": "2026-01-16T05:44:35+00:00",
                        "AttachmentId": "eni-attach-07f4a1ca86519ea5a",
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "Status": "attaching",
                        "NetworkCardIndex": 0
                    },
                    "Description": "",
                    "Groups": [
                        {
                            "GroupId": "sg-01e5657502335b671",
                            "GroupName": "free-tier-sg"
                        }
                    ],
                    "Ipv6Addresses": [],
                    "MacAddress": "0a:ff:df:ae:c0:a7",
                    "NetworkInterfaceId": "eni-0fe91ef906afd77d6",
                    "OwnerId": "129585540940",
                    "PrivateDnsName": "ip-172-31-30-193.ec2.internal",
-- More  --
```

11. Step 7 – Verify Instance Creation

```
aws ec2 describe-instances \
 --query 'Reservations[*].Instances[*].[InstanceId,State.Name,PublicIpAddress]' \
 --output table
```

Confirm:

- State = running
- Public IP assigned



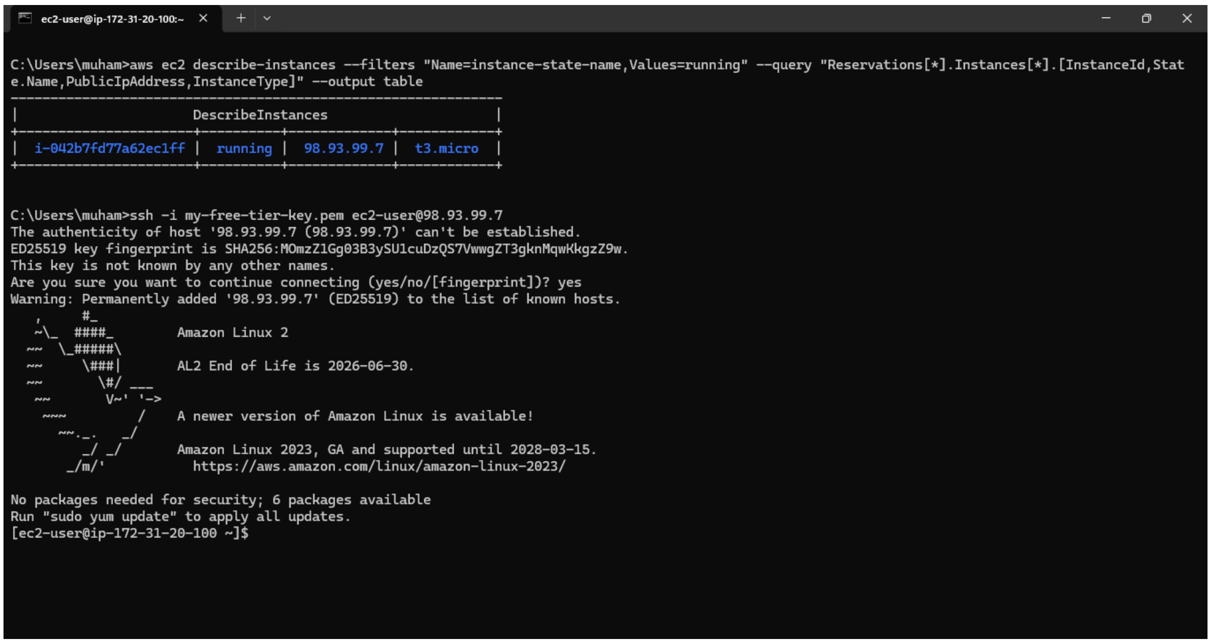
```
C:\Users\muham>aws ec2 describe-instances --query "Reservations[*].Instances[*].[InstanceId,InstanceType,State.Name,PublicIpAddress]" --output table

-----------------------------------------------------------------
|                       DescribeInstances                       |
+----------------------+------------+-----------+--------------+
|  i-0e7163042f837017e |  t3.micro  |  running  |  54.164.8.4  |
+----------------------+------------+-----------+--------------+


C:\Users\muham>
```

12. Step 8 – Access the EC2 Instance (Optional)

ssh -i my-free-tier-key.pem ec2-user@<public-ip>

Successful login confirms:

- Key pair works
- Security group rules are correct
- Instance is operational



```
C:\Users\muham>aws ec2 describe-instances --filters "Name=instance-state-name,Values=running" --query "Reservations[*].Instances[*].[InstanceId,Stat
e.Name,PublicIpAddress,InstanceType]" --output table
-----------------------------------------------------------------
|                       DescribeInstances                       |
+----------------------+-----------+-------------+-------------+
|  i-042b7fd77a62ec1ff |  running  |  98.93.99.7 |  t3.micro   |
+----------------------+-----------+-------------+-------------+


C:\Users\muham>ssh -i my-free-tier-key.pem ec2-user@98.93.99.7
The authenticity of host '98.93.99.7 (98.93.99.7)' can't be established.
ED25519 key fingerprint is SHA256:MOmzZ1Gg03B3ySU1cuDzQS7VwwgZT3gknMqwKkgzZ9w.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '98.93.99.7' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

No packages needed for security; 6 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-20-100 ~]$
```

13. Step 9 – Monitor Free Tier Usage

Free Tier Limits:

- **750 hours/month**
- Instance type: t3.micro
- OS: Amazon Linux 2

Exceeding limits may cause billing.

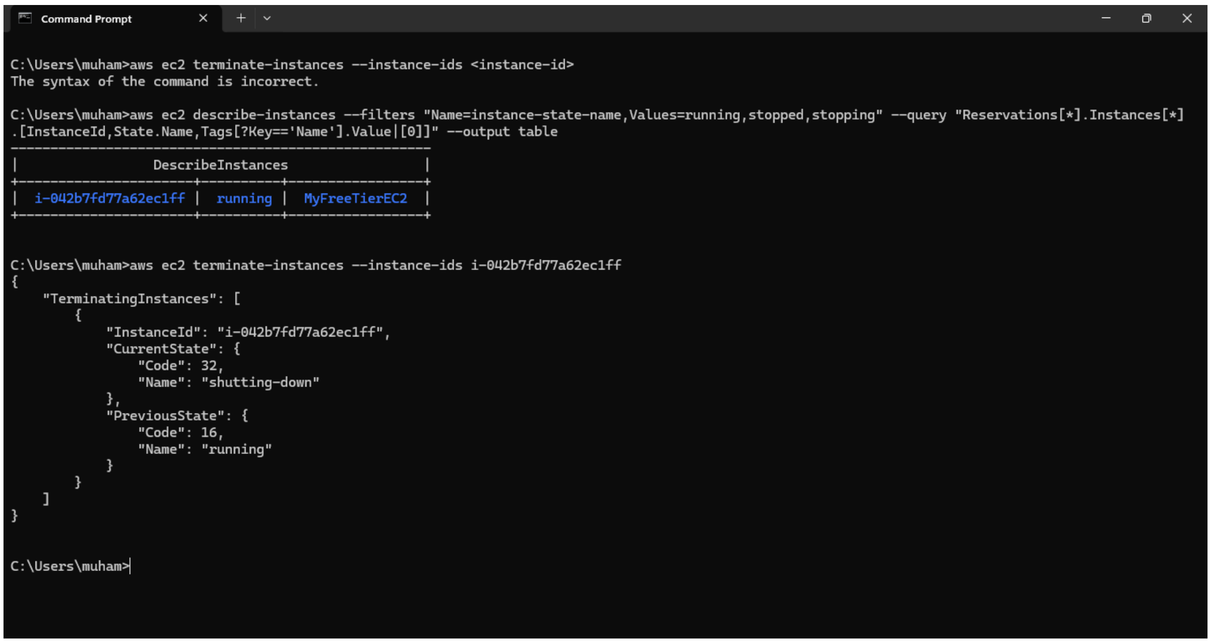
14. Step 10 – Clean Up Resources (Very Important)

14.1 Terminate EC2 Instance

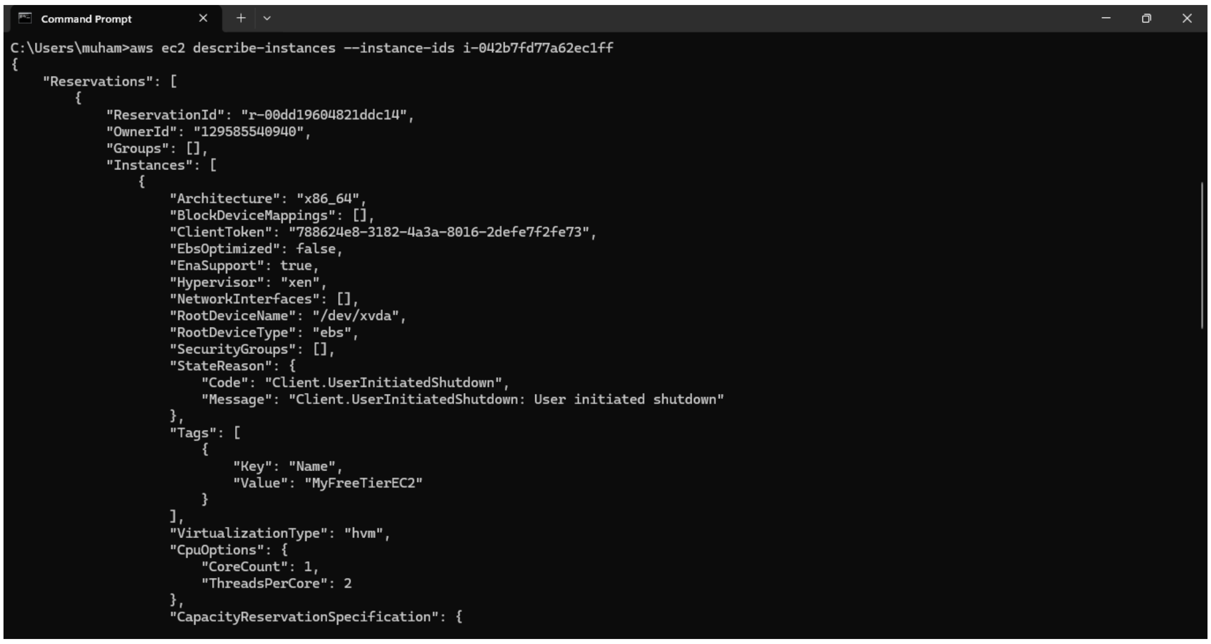
```
aws ec2 terminate-instances --instance-ids <instance-id>
```

Verify:

```
aws ec2 describe-instances --instance-ids <instance-id>
```



```
C:\Users\muham>aws ec2 terminate-instances --instance-ids <instance-id>
The syntax of the command is incorrect.

C:\Users\muham>aws ec2 describe-instances --filters "Name=instance-state-name,Values=running,stopped,stopping" --query "Reservations[*].Instances[*]
.[InstanceId,State.Name,Tags[?Key=='Name'].Value|[0]]" --output table
-----------------------------------------------------
|                 DescribeInstances                 |
+----------------------+----------+-----------------+
|  i-042b7fd77a62ec1ff |  running |  MyFreeTierEC2  |
+----------------------+----------+-----------------+


C:\Users\muham>aws ec2 terminate-instances --instance-ids i-042b7fd77a62ec1ff
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-042b7fd77a62ec1ff",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}


C:\Users\muham>
```

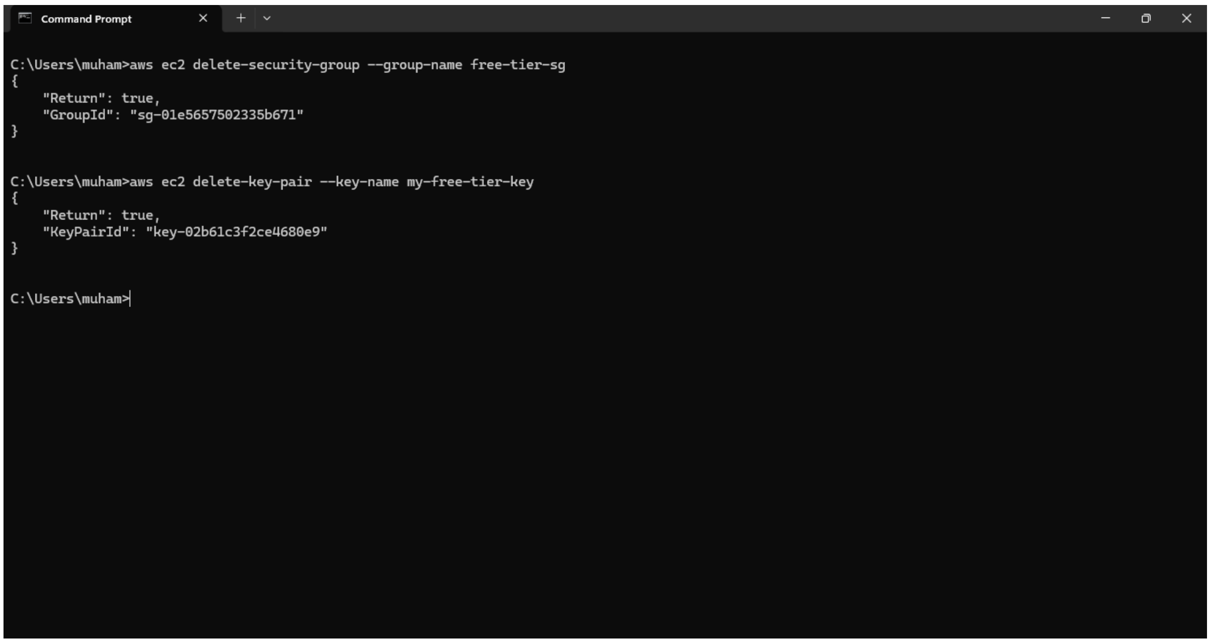


```
C:\Users\muham>aws ec2 describe-instances --instance-ids i-042b7fd77a62ec1ff
{
    "Reservations": [
        {
            "ReservationId": "r-00dd19604821ddc14",
            "OwnerId": "129585540940",
            "Groups": [],
            "Instances": [
                {
                    "Architecture": "x86_64",
                    "BlockDeviceMappings": [],
                    "ClientToken": "788624e8-3182-4a3a-8016-2defe7f2fe73",
                    "EbsOptimized": false,
                    "EnaSupport": true,
                    "Hypervisor": "xen",
                    "NetworkInterfaces": [],
                    "RootDeviceName": "/dev/xvda",
                    "RootDeviceType": "ebs",
                    "SecurityGroups": [],
                    "StateReason": {
                        "Code": "Client.UserInitiatedShutdown",
                        "Message": "Client.UserInitiatedShutdown: User initiated shutdown"
                    },
                    "Tags": [
                        {
                            "Key": "Name",
                            "Value": "MyFreeTierEC2"
                        }
                    ],
                    "VirtualizationType": "hvm",
                    "CpuOptions": {
                        "CoreCount": 1,
                        "ThreadsPerCore": 2
                    },
                    "CapacityReservationSpecification": {
```

14.2 (Optional) Delete Security Group & Key Pair

```
aws ec2 delete-security-group --group-name free-tier-sg
```

```
aws ec2 delete-key-pair --key-name my-free-tier-key
```



```
C:\Users\muham>aws ec2 delete-security-group --group-name free-tier-sg
{
    "Return": true,
    "GroupId": "sg-01e5657502335b671"
}


C:\Users\muham>aws ec2 delete-key-pair --key-name my-free-tier-key
{
    "Return": true,
    "KeyPairId": "key-02b61c3f2ce4680e9"
}


C:\Users\muham>
```

15. Challenges Faced and Solutions

Challenge 1: SSH Permission Denied Error

- Issue: Could not connect to EC2 instance after recreating key pair

- Root Cause: Old key was still configured on the running instance

- Solution: Terminated the instance and launched a fresh one with new key pair

Challenge 2: Free Tier Instance Type Error

- Issue: t2.micro showed "not eligible for Free Tier" error

- Solution: Used t3.micro instead, which is Free Tier eligible in us-east-1

Challenge 3: Security Group Configuration

- Issue: Initially forgot to allow SSH access (port 22)

- Solution: Added ingress rule using authorize-security-group-ingress command

16. Outcome

I'm successfully:

- Launched an EC2 instance using AWS CLI

- Created and managed security groups and key pairs
- Practiced real-world infrastructure provisioning
- Understood Free Tier cost control
- Followed DevOps-style cloud workflows

17. Conclusion

This hands-on lab gave me practical experience with AWS CLI and EC2 instance management. I learned how automation through command-line tools is more efficient than using the web console, especially when managing multiple resources. The main takeaway was understanding the complete lifecycle of an EC2 instance - from creation to termination - and the importance of proper cleanup to avoid unexpected charges.

The most valuable lesson was troubleshooting the SSH connection issues, which taught me about key pair management and the relationship between AWS resources. This experience will be helpful for future cloud projects and DevOps workflows.