FATIMA JINNAH WOMEN UNIVERSITY

Department of Software Engineering

LAB #09

SUBJECT: CLOUD COMPUTING

SUBMITTED TO: SIR MUHAMMAD SHOAIB

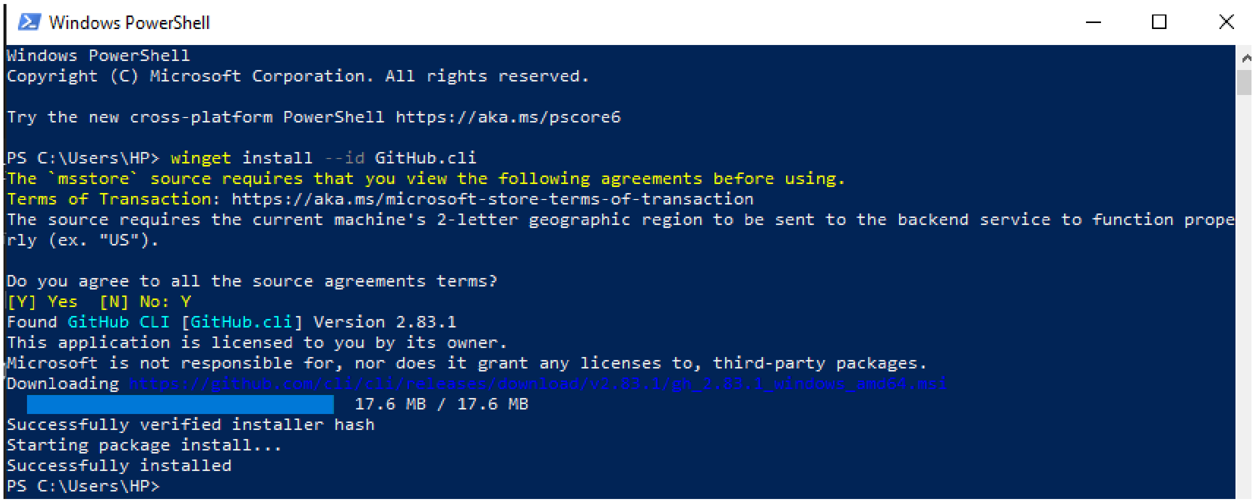
SUBMITTED BY: UMBER QASIM

REGISTRATION NO: 2023-BSE-066

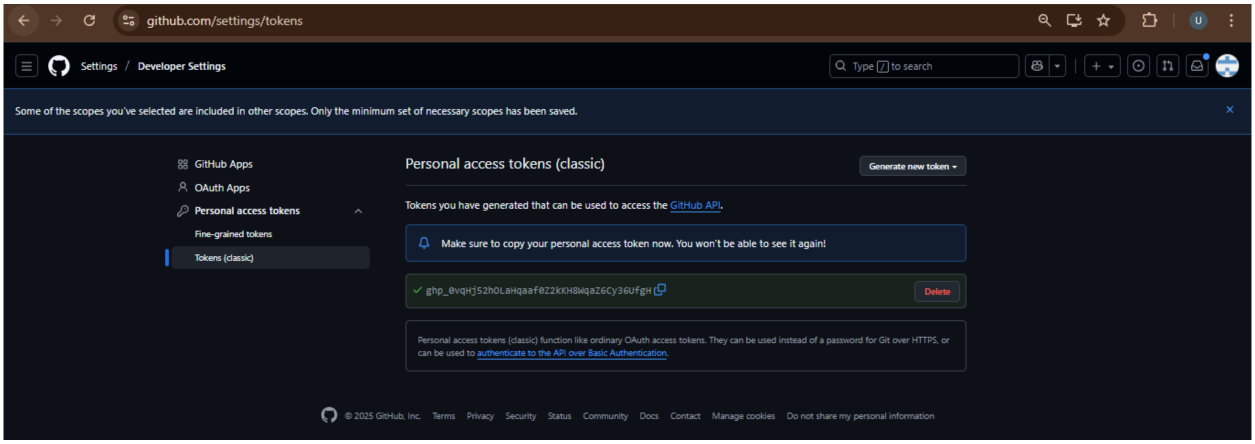
CLASS: BSSE V-B

GH CLI (Codespaces), AWS CLI, EC2, IAM, Security Groups, Filters & Queries

Task#01: GitHub CLI, Codespace setup and authentication



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\HP> winget install --id GitHub.cli
The `msstore` source requires that you view the following agreements before using.
Terms of Transaction: https://aka.ms/microsoft-store-terms-of-transaction
The source requires the current machine's 2-letter geographic region to be sent to the backend service to function prope
rly (ex. "US").

Do you agree to all the source agreements terms?
[Y] Yes  [N] No: Y
Found GitHub CLI [GitHub.cli] Version 2.83.1
This application is licensed to you by its owner.
Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.
Downloading https://github.com/cli/cli/releases/download/v2.83.1/gh_2.83.1_windows_amd64.msi
  ██████████████████████████████  17.6 MB / 17.6 MB
Successfully verified installer hash
Starting package install...
Successfully installed
PS C:\Users\HP>
```



Some of the scopes you've selected are included in other scopes. Only the minimum set of necessary scopes has been saved.

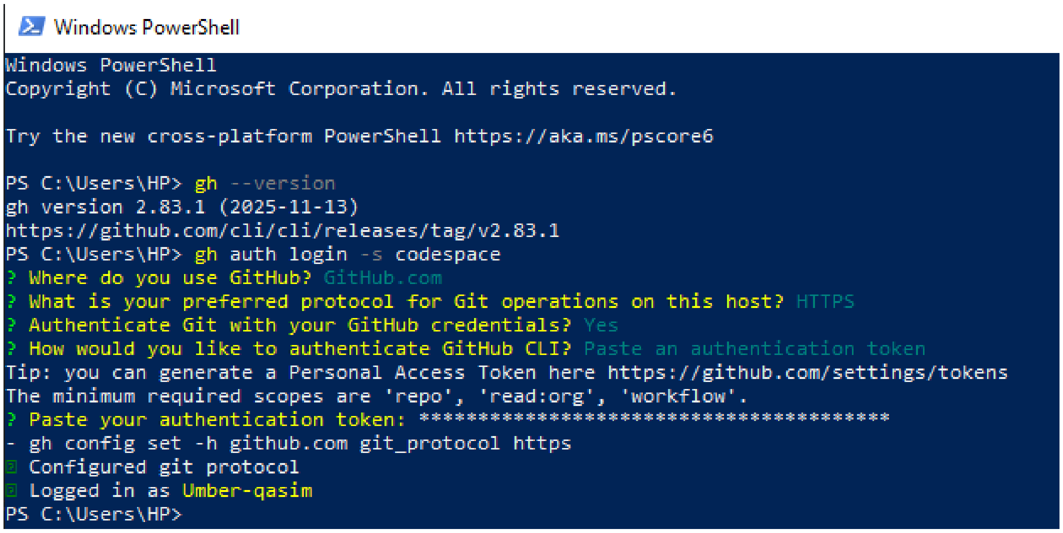
Personal access tokens (classic)

Tokens you have generated that can be used to access the GitHub API.

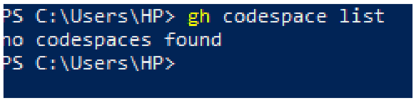
Make sure to copy your personal access token now. You won't be able to see it again!

✓ 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

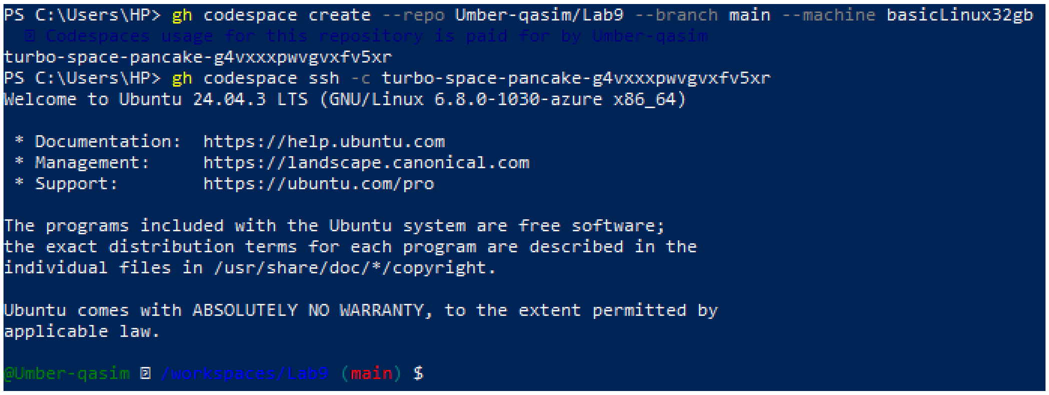
Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\HP> gh --version
gh version 2.83.1 (2025-11-13)
https://github.com/cli/cli/releases/tag/v2.83.1
PS C:\Users\HP> gh auth login -s codespace
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Paste an authentication token
Tip: you can generate a Personal Access Token here https://github.com/settings/tokens
The minimum required scopes are 'repo', 'read:org', 'workflow'.
? Paste your authentication token: ****************************************
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as Umber-qasim
PS C:\Users\HP>
```

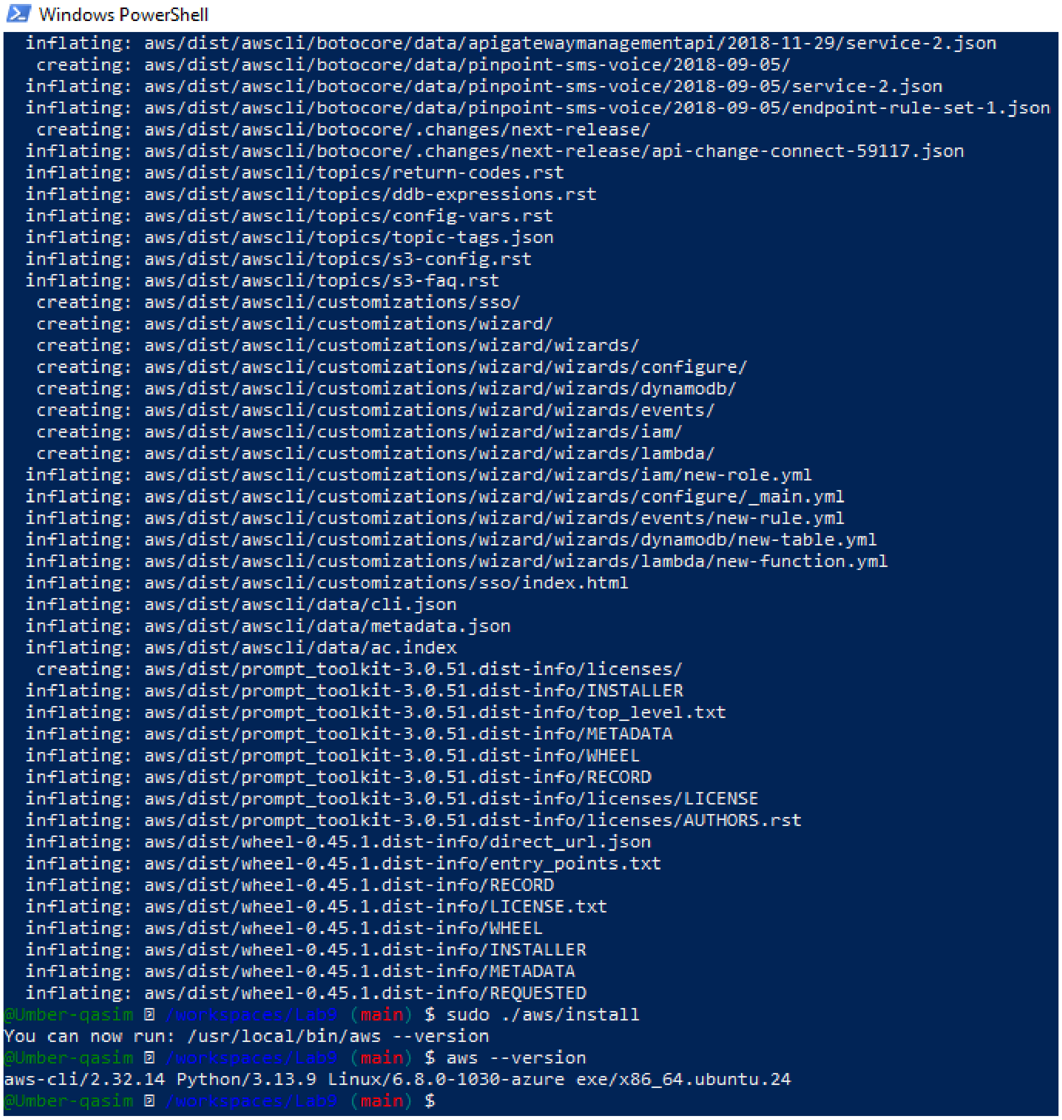


```
PS C:\Users\HP> gh codespace list
no codespaces found
PS C:\Users\HP>
```



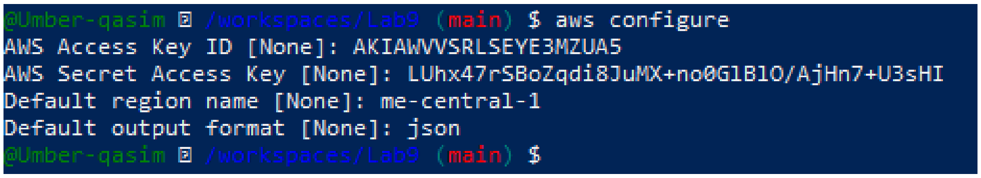
```
PS C:\Users\HP> gh codespace create --repo Umber-qasim/Lab9 --branch main --machine basicLinux32gb
  ⍰ Codespaces usage for this repository is paid for by Umber-qasim
turbo-space-pancake-g4vxxxpwvgvxfv5xr
PS C:\Users\HP> gh codespace ssh -c turbo-space-pancake-g4vxxxpwvgvxfv5xr
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-1030-azure x86_64)

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

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.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

@Umber-qasim ⍰ /workspaces/Lab9 (main) $
```

<u>Task#02:</u> Install AWS CLI inside the Codespace and configure it

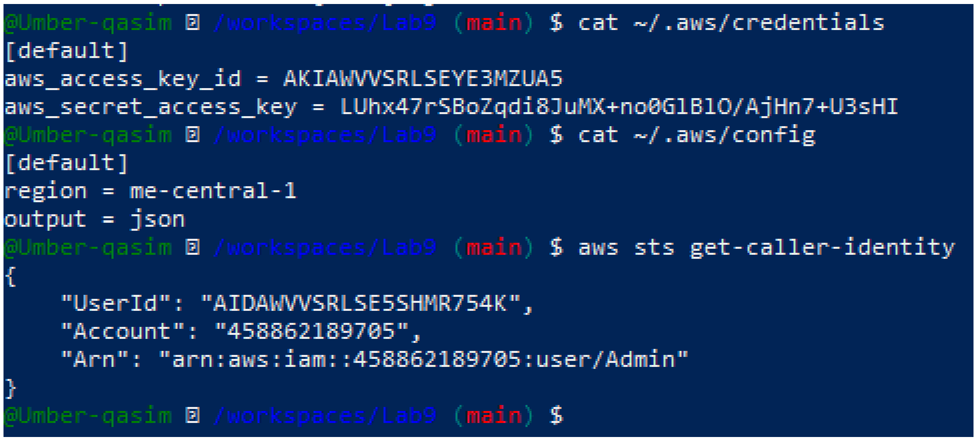


Windows PowerShell

```
  inflating: aws/dist/awscli/botocore/data/apigatewaymanagementapi/2018-11-29/service-2.json
   creating: aws/dist/awscli/botocore/data/pinpoint-sms-voice/2018-09-05/
  inflating: aws/dist/awscli/botocore/data/pinpoint-sms-voice/2018-09-05/service-2.json
  inflating: aws/dist/awscli/botocore/data/pinpoint-sms-voice/2018-09-05/endpoint-rule-set-1.json
   creating: aws/dist/awscli/botocore/.changes/next-release/
  inflating: aws/dist/awscli/botocore/.changes/next-release/api-change-connect-59117.json
  inflating: aws/dist/awscli/topics/return-codes.rst
  inflating: aws/dist/awscli/topics/ddb-expressions.rst
  inflating: aws/dist/awscli/topics/config-vars.rst
  inflating: aws/dist/awscli/topics/topic-tags.json
  inflating: aws/dist/awscli/topics/s3-config.rst
  inflating: aws/dist/awscli/topics/s3-faq.rst
   creating: aws/dist/awscli/customizations/sso/
   creating: aws/dist/awscli/customizations/wizard/
   creating: aws/dist/awscli/customizations/wizard/wizards/
   creating: aws/dist/awscli/customizations/wizard/wizards/configure/
   creating: aws/dist/awscli/customizations/wizard/wizards/dynamodb/
   creating: aws/dist/awscli/customizations/wizard/wizards/events/
   creating: aws/dist/awscli/customizations/wizard/wizards/iam/
   creating: aws/dist/awscli/customizations/wizard/wizards/lambda/
  inflating: aws/dist/awscli/customizations/wizard/wizards/iam/new-role.yml
  inflating: aws/dist/awscli/customizations/wizard/wizards/configure/_main.yml
  inflating: aws/dist/awscli/customizations/wizard/wizards/events/new-rule.yml
  inflating: aws/dist/awscli/customizations/wizard/wizards/dynamodb/new-table.yml
  inflating: aws/dist/awscli/customizations/wizard/wizards/lambda/new-function.yml
  inflating: aws/dist/awscli/customizations/sso/index.html
  inflating: aws/dist/awscli/data/cli.json
  inflating: aws/dist/awscli/data/metadata.json
  inflating: aws/dist/awscli/data/ac.index
   creating: aws/dist/prompt_toolkit-3.0.51.dist-info/licenses/
  inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/INSTALLER
  inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/top_level.txt
  inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/METADATA
  inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/WHEEL
  inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/RECORD
  inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/licenses/LICENSE
  inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/licenses/AUTHORS.rst
  inflating: aws/dist/wheel-0.45.1.dist-info/direct_url.json
  inflating: aws/dist/wheel-0.45.1.dist-info/entry_points.txt
  inflating: aws/dist/wheel-0.45.1.dist-info/RECORD
  inflating: aws/dist/wheel-0.45.1.dist-info/LICENSE.txt
  inflating: aws/dist/wheel-0.45.1.dist-info/WHEEL
  inflating: aws/dist/wheel-0.45.1.dist-info/INSTALLER
  inflating: aws/dist/wheel-0.45.1.dist-info/METADATA
  inflating: aws/dist/wheel-0.45.1.dist-info/REQUESTED
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ sudo ./aws/install
You can now run: /usr/local/bin/aws --version
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws --version
aws-cli/2.32.14 Python/3.13.9 Linux/6.8.0-1030-azure exe/x86_64.ubuntu.24
@Umber-qasim ⍰ /workspaces/Lab9 (main) $
```

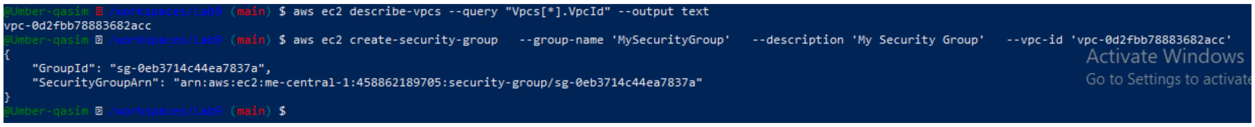


```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws configure
AWS Access Key ID [None]: AKIAWVVSRLSEYE3MZUA5
AWS Secret Access Key [None]: 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
Default region name [None]: me-central-1
Default output format [None]: json
@Umber-qasim ⊡ /workspaces/Lab9 (main) $
```

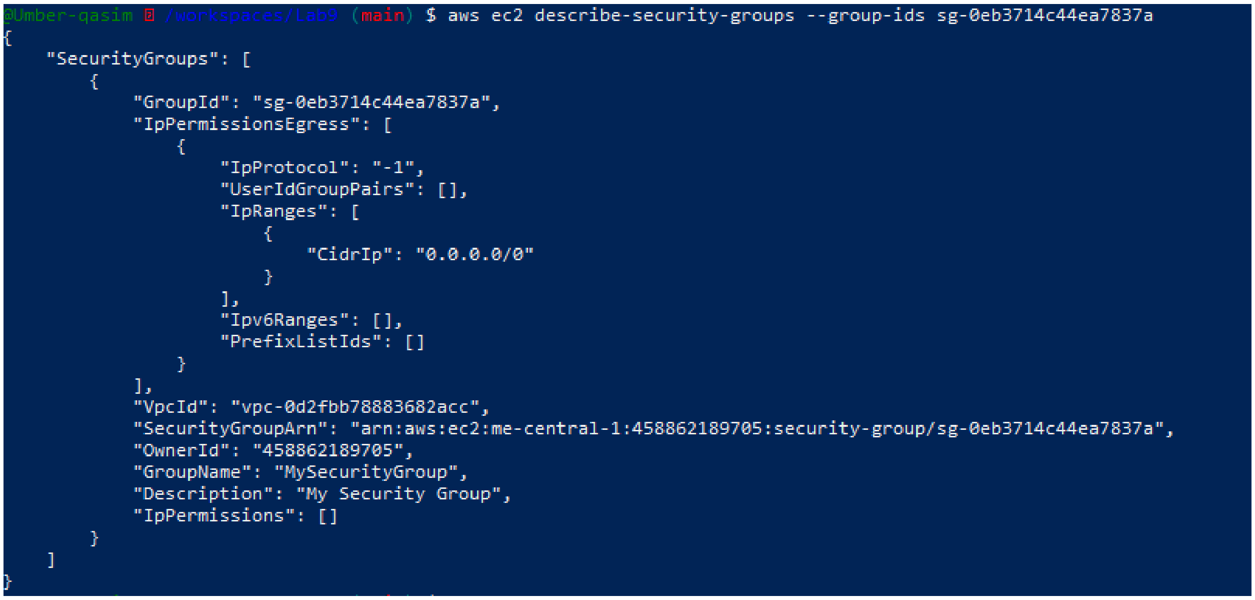


```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ cat ~/.aws/credentials
[default]
aws_access_key_id = AKIAWVVSRLSEYE3MZUA5
aws_secret_access_key = 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
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ cat ~/.aws/config
[default]
region = me-central-1
output = json
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws sts get-caller-identity
{
    "UserId": "AIDAWVVSRLSE5SHMR754K",
    "Account": "458862189705",
    "Arn": "arn:aws:iam::458862189705:user/Admin"
}
@Umber-qasim ⊡ /workspaces/Lab9 (main) $
```

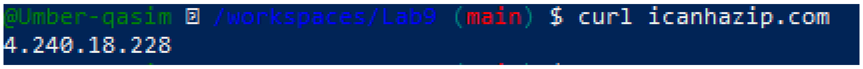
Task#03: Create security group and add ingress rules using Codespace IP



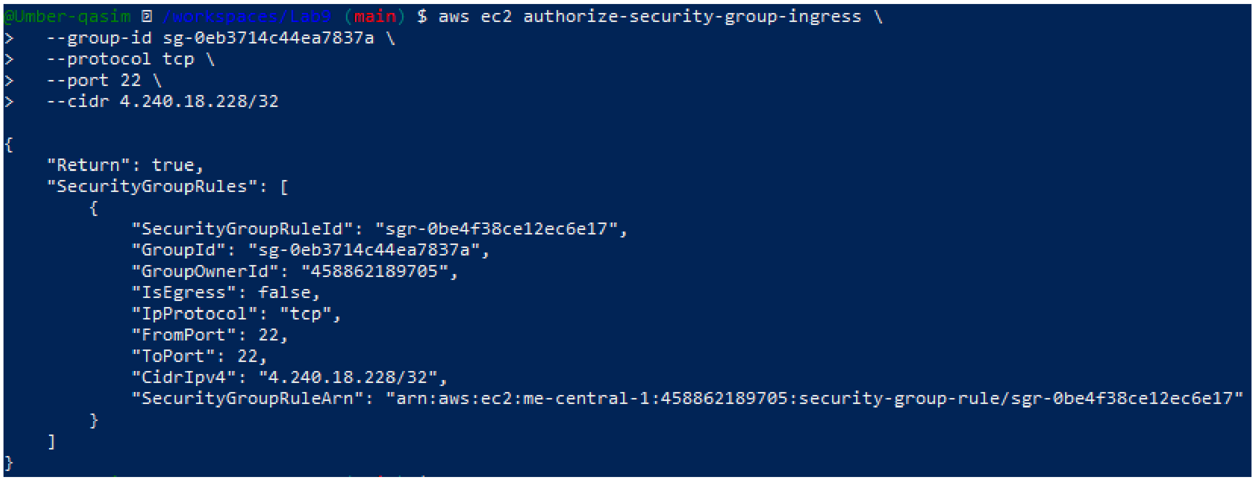
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws ec2 describe-vpcs --query "Vpcs[*].VpcId" --output text
vpc-0d2fbb78883682acc
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws ec2 create-security-group   --group-name 'MySecurityGroup'   --description 'My Security Group'   --vpc-id 'vpc-0d2fbb78883682acc'
{
    "GroupId": "sg-0eb3714c44ea7837a",
    "SecurityGroupArn": "arn:aws:ec2:me-central-1:458862189705:security-group/sg-0eb3714c44ea7837a"
}
@Umber-qasim ⊡ /workspaces/Lab9 (main) $
```



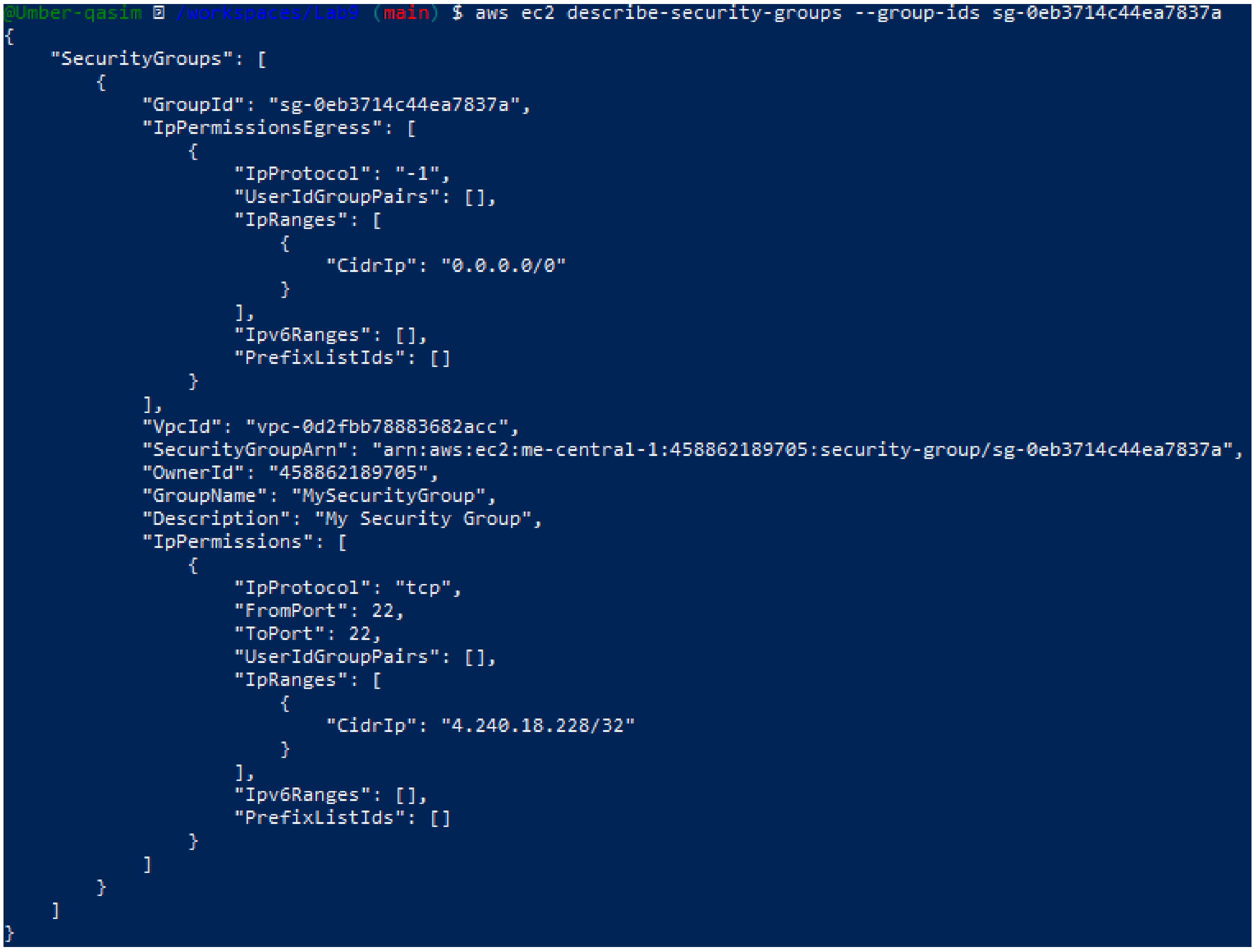
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws ec2 describe-security-groups --group-ids sg-0eb3714c44ea7837a
{
    "SecurityGroups": [
        {
            "GroupId": "sg-0eb3714c44ea7837a",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-0d2fbb78883682acc",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:458862189705:security-group/sg-0eb3714c44ea7837a",
            "OwnerId": "458862189705",
            "GroupName": "MySecurityGroup",
            "Description": "My Security Group",
            "IpPermissions": []
        }
    ]
}
```



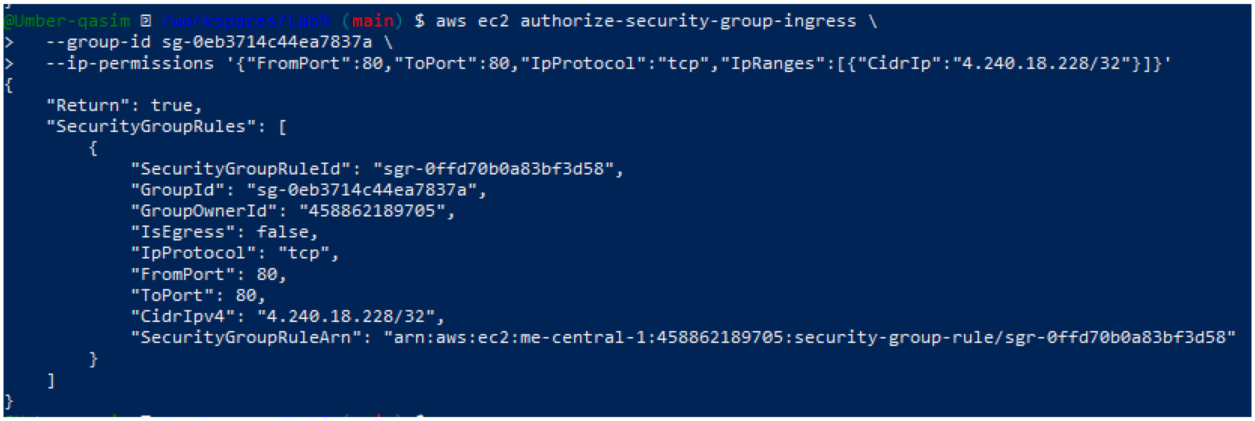
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ curl icanhazip.com
4.240.18.228
```



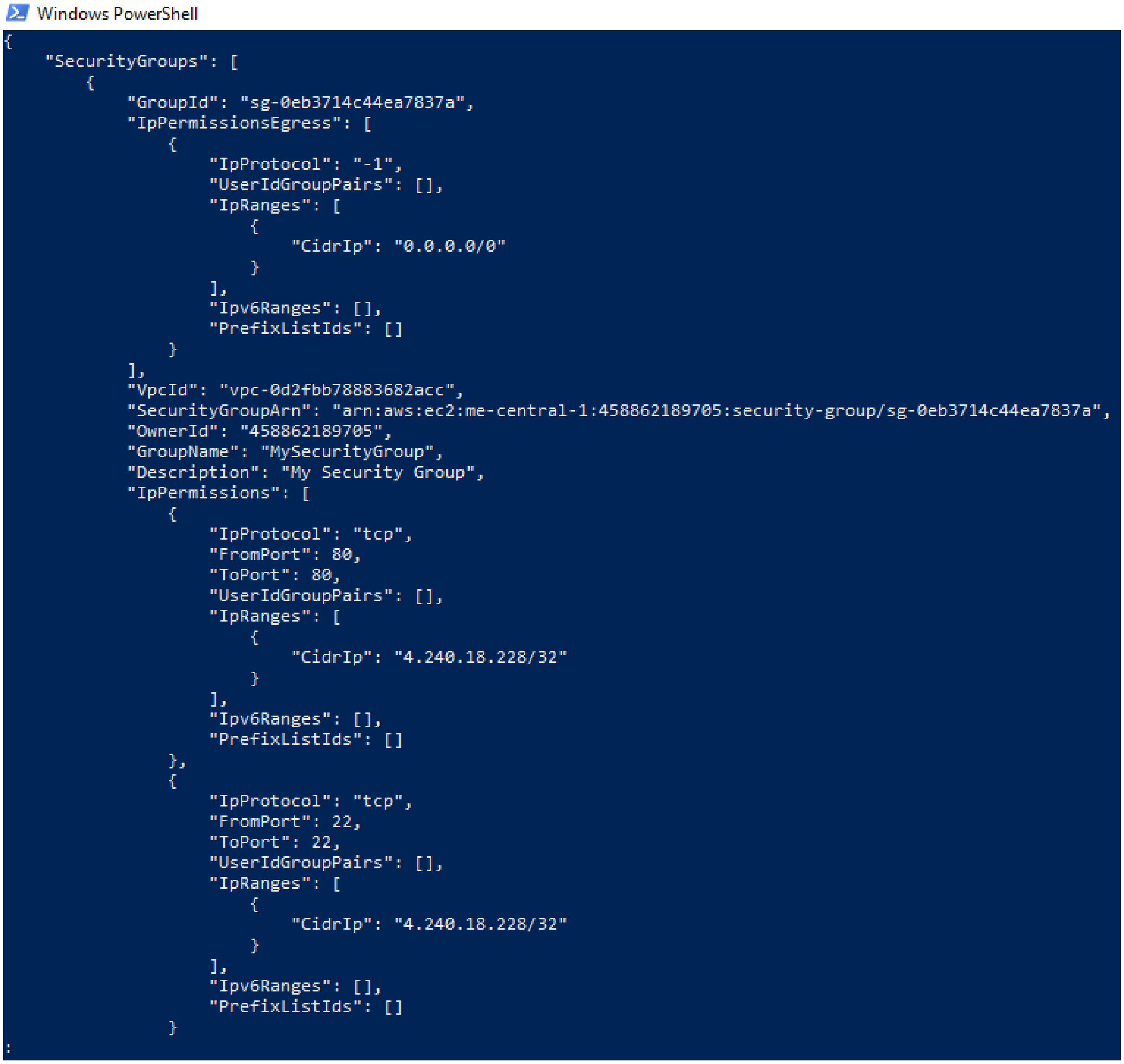
```
@Umber-qasim ⌧ /workspaces/Lab9 (main) $ aws ec2 authorize-security-group-ingress \
>   --group-id sg-0eb3714c44ea7837a \
>   --protocol tcp \
>   --port 22 \
>   --cidr 4.240.18.228/32

{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0be4f38ce12ec6e17",
            "GroupId": "sg-0eb3714c44ea7837a",
            "GroupOwnerId": "458862189705",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "4.240.18.228/32",
            "SecurityGroupRuleArn": "arn:aws:ec2:me-central-1:458862189705:security-group-rule/sgr-0be4f38ce12ec6e17"
        }
    ]
}
```



```
@Umber-qasim ⌧ /workspaces/Lab9 (main) $ aws ec2 describe-security-groups --group-ids sg-0eb3714c44ea7837a
{
    "SecurityGroups": [
        {
            "GroupId": "sg-0eb3714c44ea7837a",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-0d2fbb78883682acc",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:458862189705:security-group/sg-0eb3714c44ea7837a",
            "OwnerId": "458862189705",
            "GroupName": "MySecurityGroup",
            "Description": "My Security Group",
            "IpPermissions": [
                {
                    "IpProtocol": "tcp",
                    "FromPort": 22,
                    "ToPort": 22,
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "4.240.18.228/32"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ]
        }
    ]
}
```



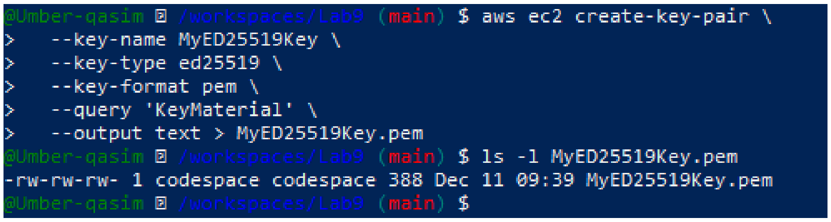
```
@Umber-qasim ➜ /workspaces/Lab9 (main) $ aws ec2 authorize-security-group-ingress \
>   --group-id sg-0eb3714c44ea7837a \
>   --ip-permissions '{"FromPort":80,"ToPort":80,"IpProtocol":"tcp","IpRanges":[{"CidrIp":"4.240.18.228/32"}]}'
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0ffd70b0a83bf3d58",
            "GroupId": "sg-0eb3714c44ea7837a",
            "GroupOwnerId": "458862189705",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "4.240.18.228/32",
            "SecurityGroupRuleArn": "arn:aws:ec2:me-central-1:458862189705:security-group-rule/sgr-0ffd70b0a83bf3d58"
        }
    ]
}
```



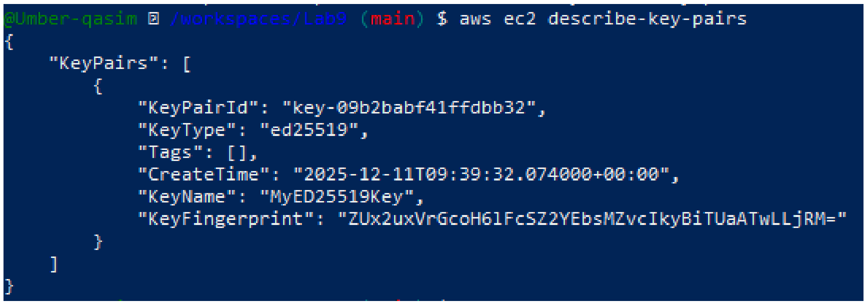
Windows PowerShell

```
{
    "SecurityGroups": [
        {
            "GroupId": "sg-0eb3714c44ea7837a",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-0d2fbb78883682acc",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:458862189705:security-group/sg-0eb3714c44ea7837a",
            "OwnerId": "458862189705",
            "GroupName": "MySecurityGroup",
            "Description": "My Security Group",
            "IpPermissions": [
                {
                    "IpProtocol": "tcp",
                    "FromPort": 80,
                    "ToPort": 80,
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "4.240.18.228/32"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                },
                {
                    "IpProtocol": "tcp",
                    "FromPort": 22,
                    "ToPort": 22,
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "4.240.18.228/32"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
:
```

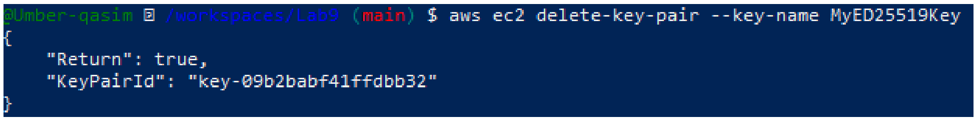
Task#04: Create a key pair, describe key pairs, and launch EC2 instance



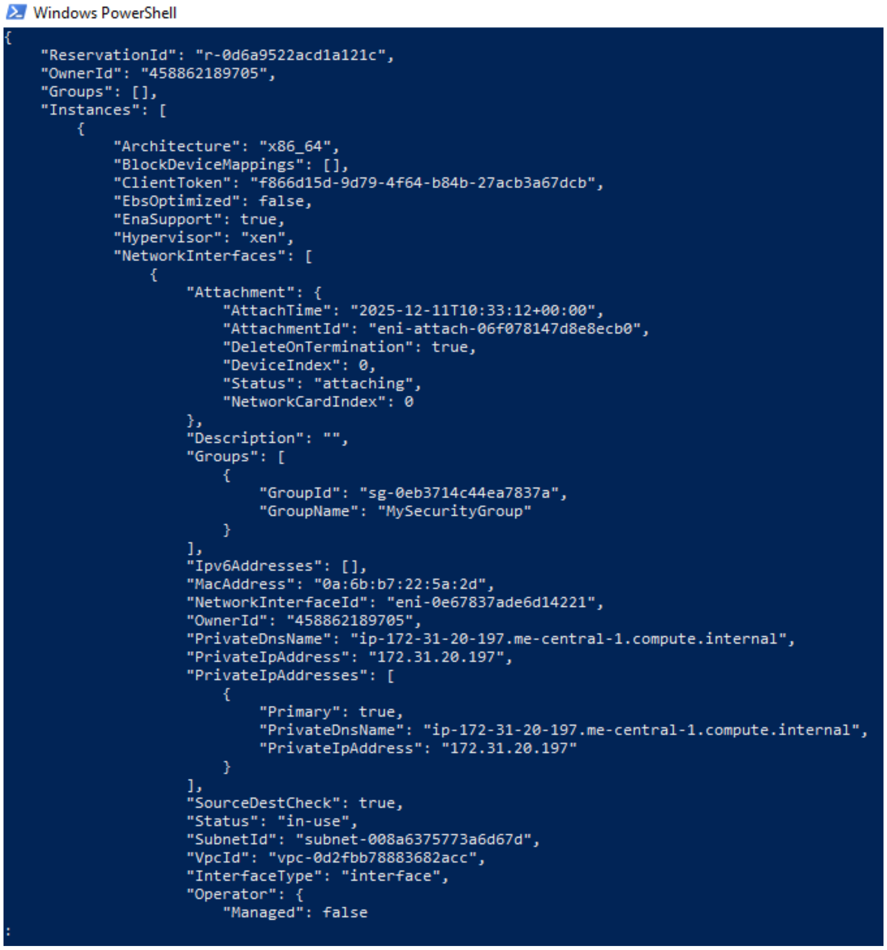
```
@Umber-qasim ⊠ /workspaces/Lab9 (main) $ aws ec2 create-key-pair \
>   --key-name MyED25519Key \
>   --key-type ed25519 \
>   --key-format pem \
>   --query 'KeyMaterial' \
>   --output text > MyED25519Key.pem
@Umber-qasim ⊠ /workspaces/Lab9 (main) $ ls -l MyED25519Key.pem
-rw-rw-rw- 1 codespace codespace 388 Dec 11 09:39 MyED25519Key.pem
@Umber-qasim ⊠ /workspaces/Lab9 (main) $
```



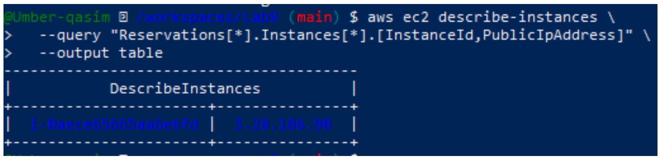
```
@Umber-qasim ⊠ /workspaces/Lab9 (main) $ aws ec2 describe-key-pairs
{
    "KeyPairs": [
        {
            "KeyPairId": "key-09b2babf41ffdbb32",
            "KeyType": "ed25519",
            "Tags": [],
            "CreateTime": "2025-12-11T09:39:32.074000+00:00",
            "KeyName": "MyED25519Key",
            "KeyFingerprint": "ZUx2uxVrGcoH6lFcSZ2YEbsMZvcIkyBiTUaATwLLjRM="
        }
    ]
}
```



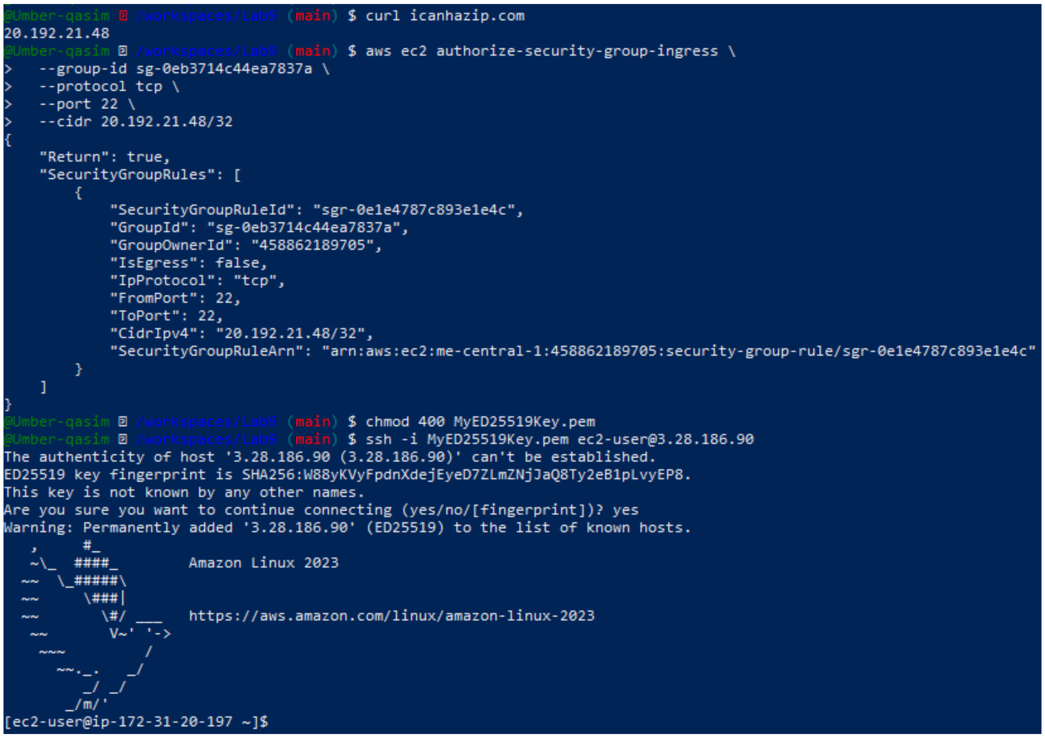
```
@Umber-qasim ⊠ /workspaces/Lab9 (main) $ aws ec2 delete-key-pair --key-name MyED25519Key
{
    "Return": true,
    "KeyPairId": "key-09b2babf41ffdbb32"
}
```



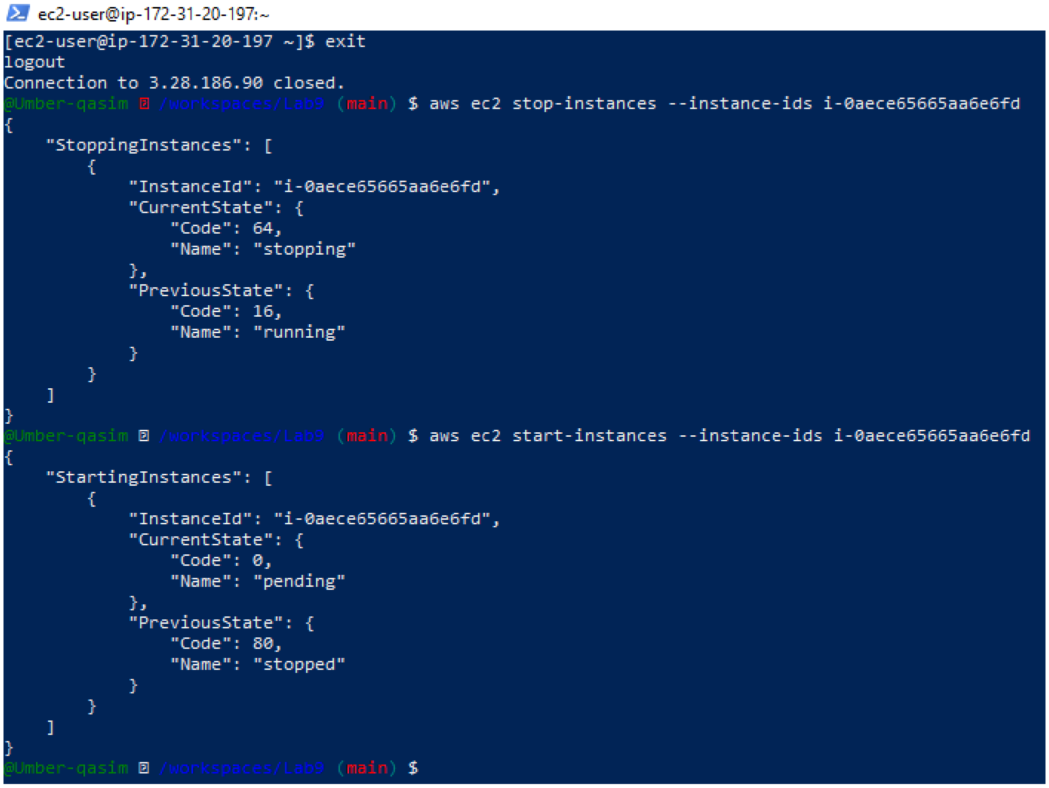
```
{
    "ReservationId": "r-0d6a9522acd1a121c",
    "OwnerId": "458862189705",
    "Groups": [],
    "Instances": [
        {
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "f866d15d-9d79-4f64-b84b-27acb3a67dcb",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
                        "AttachTime": "2025-12-11T10:33:12+00:00",
                        "AttachmentId": "eni-attach-06f078147d8e8ecb0",
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "Status": "attaching",
                        "NetworkCardIndex": 0
                    },
                    "Description": "",
                    "Groups": [
                        {
                            "GroupId": "sg-0eb3714c44ea7837a",
                            "GroupName": "MySecurityGroup"
                        }
                    ],
                    "Ipv6Addresses": [],
                    "MacAddress": "0a:6b:b7:22:5a:2d",
                    "NetworkInterfaceId": "eni-0e67837ade6d14221",
                    "OwnerId": "458862189705",
                    "PrivateDnsName": "ip-172-31-20-197.me-central-1.compute.internal",
                    "PrivateIpAddress": "172.31.20.197",
                    "PrivateIpAddresses": [
                        {
                            "Primary": true,
                            "PrivateDnsName": "ip-172-31-20-197.me-central-1.compute.internal",
                            "PrivateIpAddress": "172.31.20.197"
                        }
                    ],
                    "SourceDestCheck": true,
                    "Status": "in-use",
                    "SubnetId": "subnet-008a6375773a6d67d",
                    "VpcId": "vpc-0d2fbb78883682acc",
                    "InterfaceType": "interface",
                    "Operator": {
                        "Managed": false
:
```



```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-instances \
>   --query "Reservations[*].Instances[*].[InstanceId,PublicIpAddress]" \
>   --output table
----------------------------------------
|           DescribeInstances           |
+-----------------------+---------------+
|  i-0aece65665aa6e6fd |  3.28.186.90  |
+-----------------------+---------------+
```



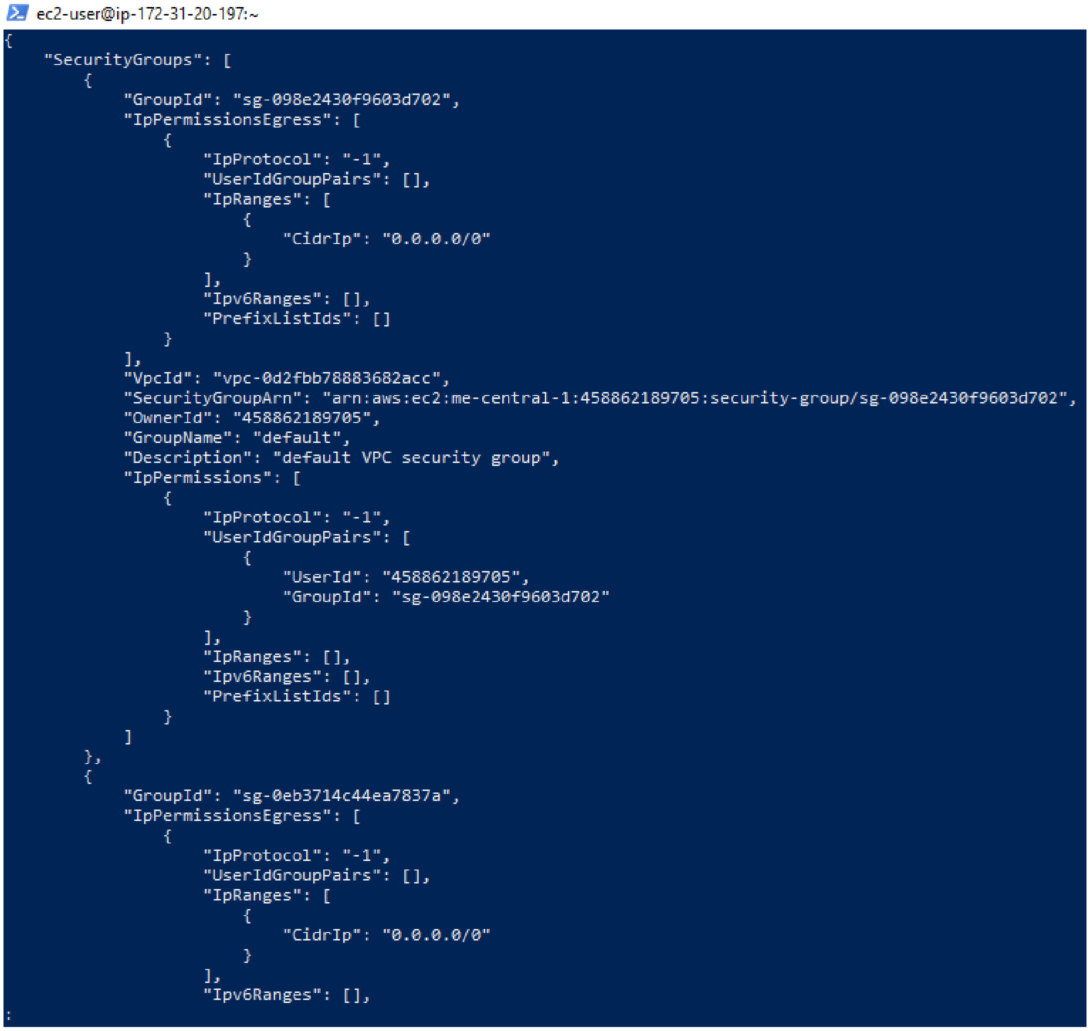
```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ curl icanhazip.com
20.192.21.48
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 authorize-security-group-ingress \
>   --group-id sg-0eb3714c44ea7837a \
>   --protocol tcp \
>   --port 22 \
>   --cidr 20.192.21.48/32
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0e1e4787c893e1e4c",
            "GroupId": "sg-0eb3714c44ea7837a",
            "GroupOwnerId": "458862189705",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "20.192.21.48/32",
            "SecurityGroupRuleArn": "arn:aws:ec2:me-central-1:458862189705:security-group-rule/sgr-0e1e4787c893e1e4c"
        }
    ]
}
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ chmod 400 MyED25519Key.pem
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ ssh -i MyED25519Key.pem ec2-user@3.28.186.90
The authenticity of host '3.28.186.90 (3.28.186.90)' can't be established.
ED25519 key fingerprint is SHA256:W88yKVyFpdnXdejEyeD7ZLmZNjJaQ8Ty2eB1pLvyEP8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.28.186.90' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-20-197 ~]$
```



```
[ec2-user@ip-172-31-20-197 ~]$ exit
logout
Connection to 3.28.186.90 closed.
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 stop-instances --instance-ids i-0aece65665aa6e6fd
{
    "StoppingInstances": [
        {
            "InstanceId": "i-0aece65665aa6e6fd",
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 start-instances --instance-ids i-0aece65665aa6e6fd
{
    "StartingInstances": [
        {
            "InstanceId": "i-0aece65665aa6e6fd",
            "CurrentState": {
                "Code": 0,
                "Name": "pending"
            },
            "PreviousState": {
                "Code": 80,
                "Name": "stopped"
            }
        }
    ]
}
@Umber-qasim ⍰ /workspaces/Lab9 (main) $
```

Task#05: Understand AWS describe-* commands

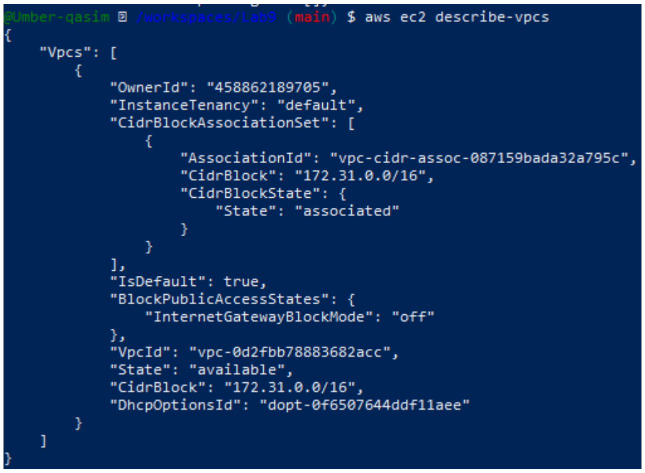
Security Groups



ec2-user@ip-172-31-20-197:~

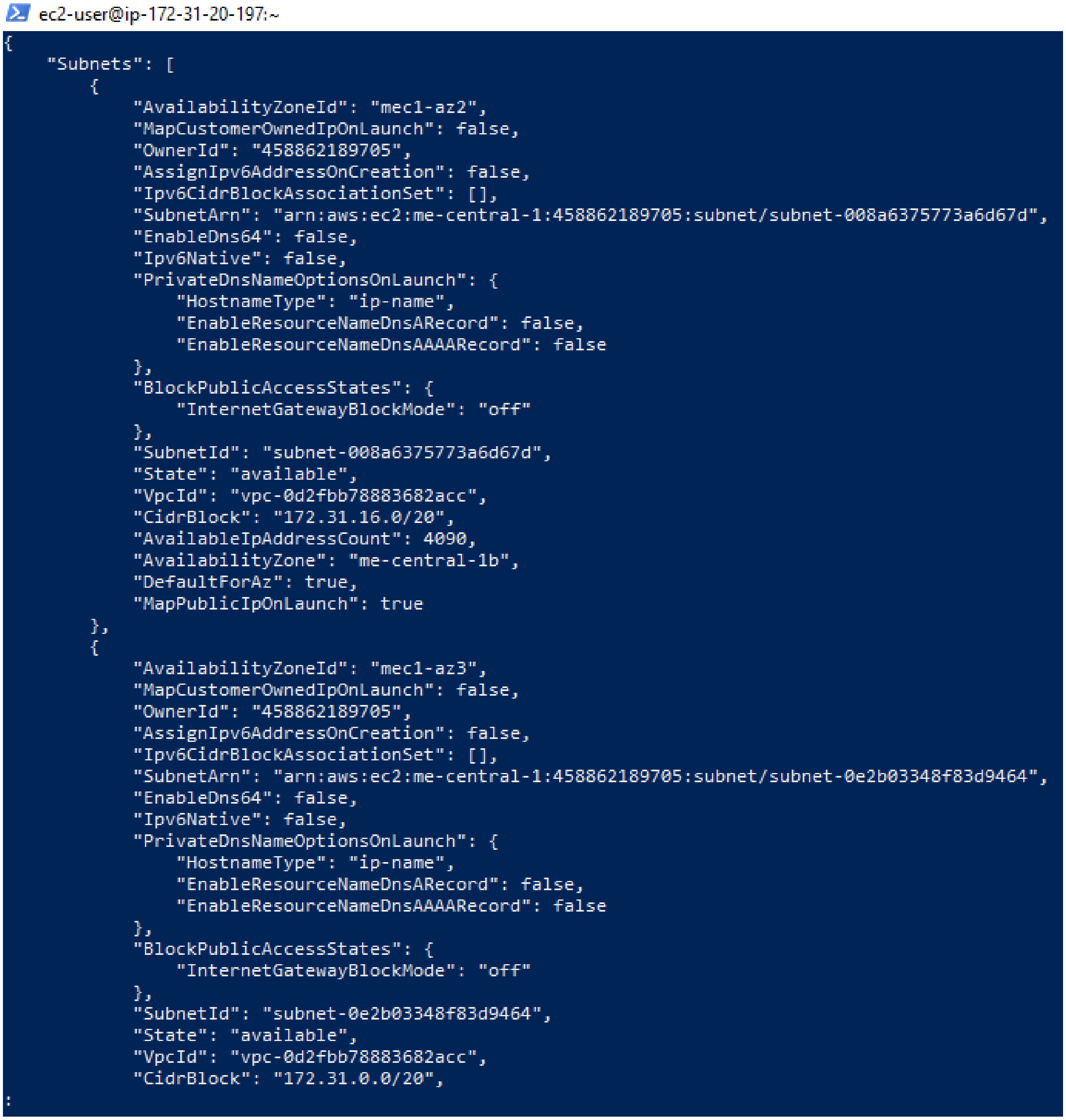
```
{
    "SecurityGroups": [
        {
            "GroupId": "sg-098e2430f9603d702",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-0d2fbb78883682acc",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:458862189705:security-group/sg-098e2430f9603d702",
            "OwnerId": "458862189705",
            "GroupName": "default",
            "Description": "default VPC security group",
            "IpPermissions": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [
                        {
                            "UserId": "458862189705",
                            "GroupId": "sg-098e2430f9603d702"
                        }
                    ],
                    "IpRanges": [],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ]
        },
        {
            "GroupId": "sg-0eb3714c44ea7837a",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
:
```

VPCs



```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-vpcs
{
    "Vpcs": [
        {
            "OwnerId": "458862189705",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-087159bada32a795c",
                    "CidrBlock": "172.31.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": true,
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "VpcId": "vpc-0d2fbb78883682acc",
            "State": "available",
            "CidrBlock": "172.31.0.0/16",
            "DhcpOptionsId": "dopt-0f6507644ddf11aee"
        }
    ]
}
```

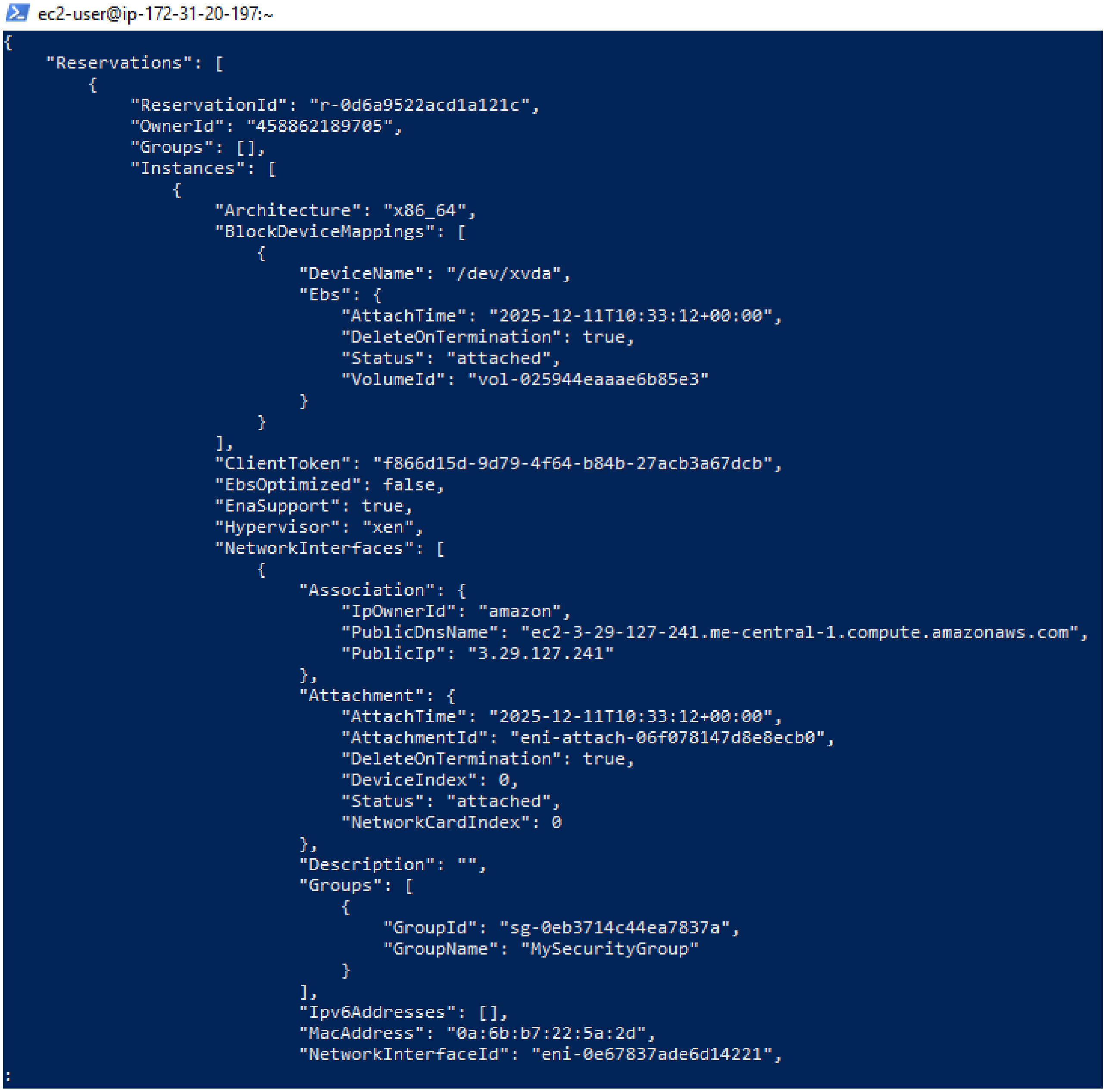
Subnets



ec2-user@ip-172-31-20-197:~

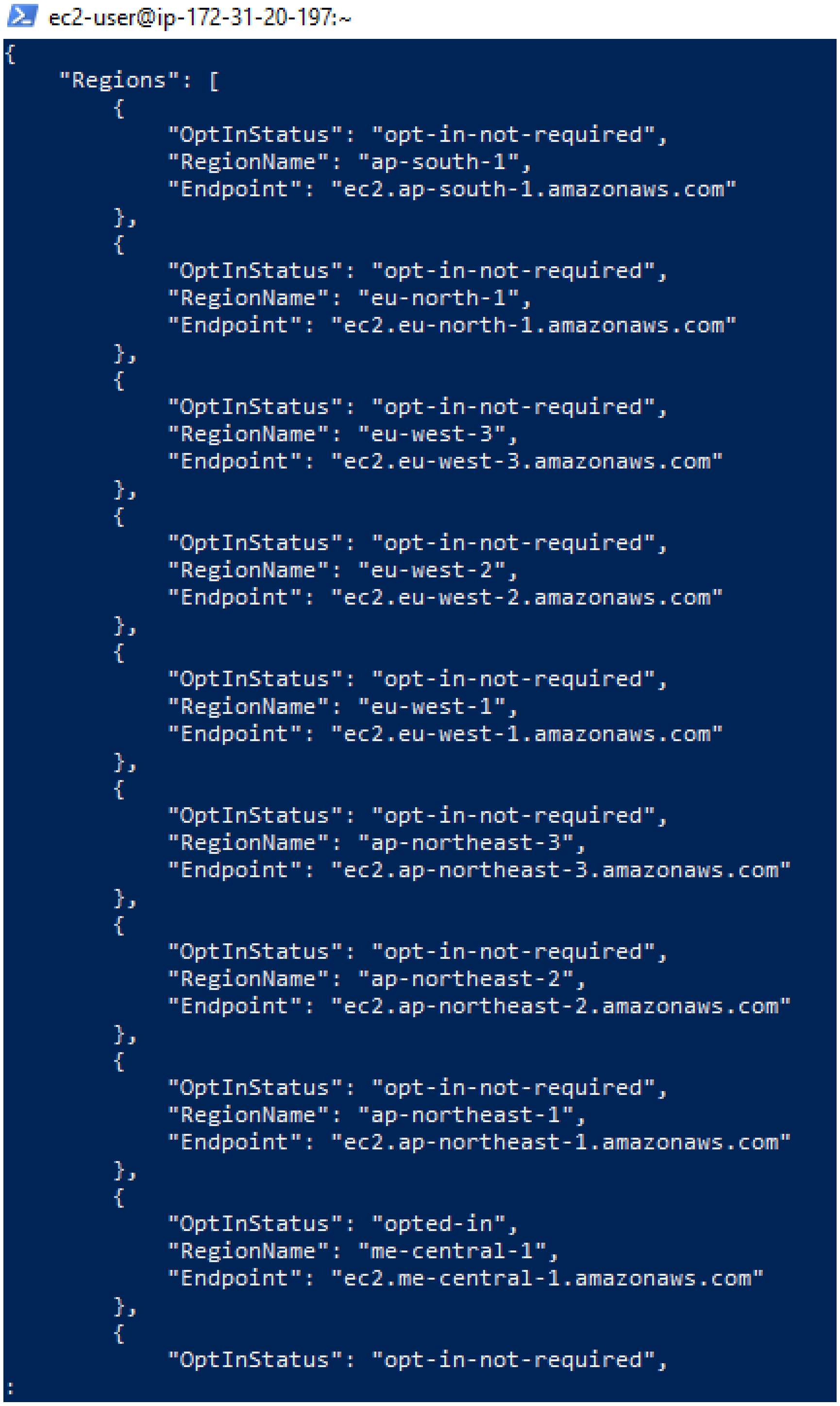
```
{
    "Subnets": [
        {
            "AvailabilityZoneId": "mec1-az2",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "458862189705",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "SubnetArn": "arn:aws:ec2:me-central-1:458862189705:subnet/subnet-008a6375773a6d67d",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            },
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "SubnetId": "subnet-008a6375773a6d67d",
            "State": "available",
            "VpcId": "vpc-0d2fbb78883682acc",
            "CidrBlock": "172.31.16.0/20",
            "AvailableIpAddressCount": 4090,
            "AvailabilityZone": "me-central-1b",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true
        },
        {
            "AvailabilityZoneId": "mec1-az3",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "458862189705",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "SubnetArn": "arn:aws:ec2:me-central-1:458862189705:subnet/subnet-0e2b03348f83d9464",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            },
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "SubnetId": "subnet-0e2b03348f83d9464",
            "State": "available",
            "VpcId": "vpc-0d2fbb78883682acc",
            "CidrBlock": "172.31.0.0/20",
:
```

Instances



```
{
    "Reservations": [
        {
            "ReservationId": "r-0d6a9522acd1a121c",
            "OwnerId": "458862189705",
            "Groups": [],
            "Instances": [
                {
                    "Architecture": "x86_64",
                    "BlockDeviceMappings": [
                        {
                            "DeviceName": "/dev/xvda",
                            "Ebs": {
                                "AttachTime": "2025-12-11T10:33:12+00:00",
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-025944eaaae6b85e3"
                            }
                        }
                    ],
                    "ClientToken": "f866d15d-9d79-4f64-b84b-27acb3a67dcb",
                    "EbsOptimized": false,
                    "EnaSupport": true,
                    "Hypervisor": "xen",
                    "NetworkInterfaces": [
                        {
                            "Association": {
                                "IpOwnerId": "amazon",
                                "PublicDnsName": "ec2-3-29-127-241.me-central-1.compute.amazonaws.com",
                                "PublicIp": "3.29.127.241"
                            },
                            "Attachment": {
                                "AttachTime": "2025-12-11T10:33:12+00:00",
                                "AttachmentId": "eni-attach-06f078147d8e8ecb0",
                                "DeleteOnTermination": true,
                                "DeviceIndex": 0,
                                "Status": "attached",
                                "NetworkCardIndex": 0
                            },
                            "Description": "",
                            "Groups": [
                                {
                                    "GroupId": "sg-0eb3714c44ea7837a",
                                    "GroupName": "MySecurityGroup"
                                }
                            ],
                            "Ipv6Addresses": [],
                            "MacAddress": "0a:6b:b7:22:5a:2d",
                            "NetworkInterfaceId": "eni-0e67837ade6d14221",
:
```

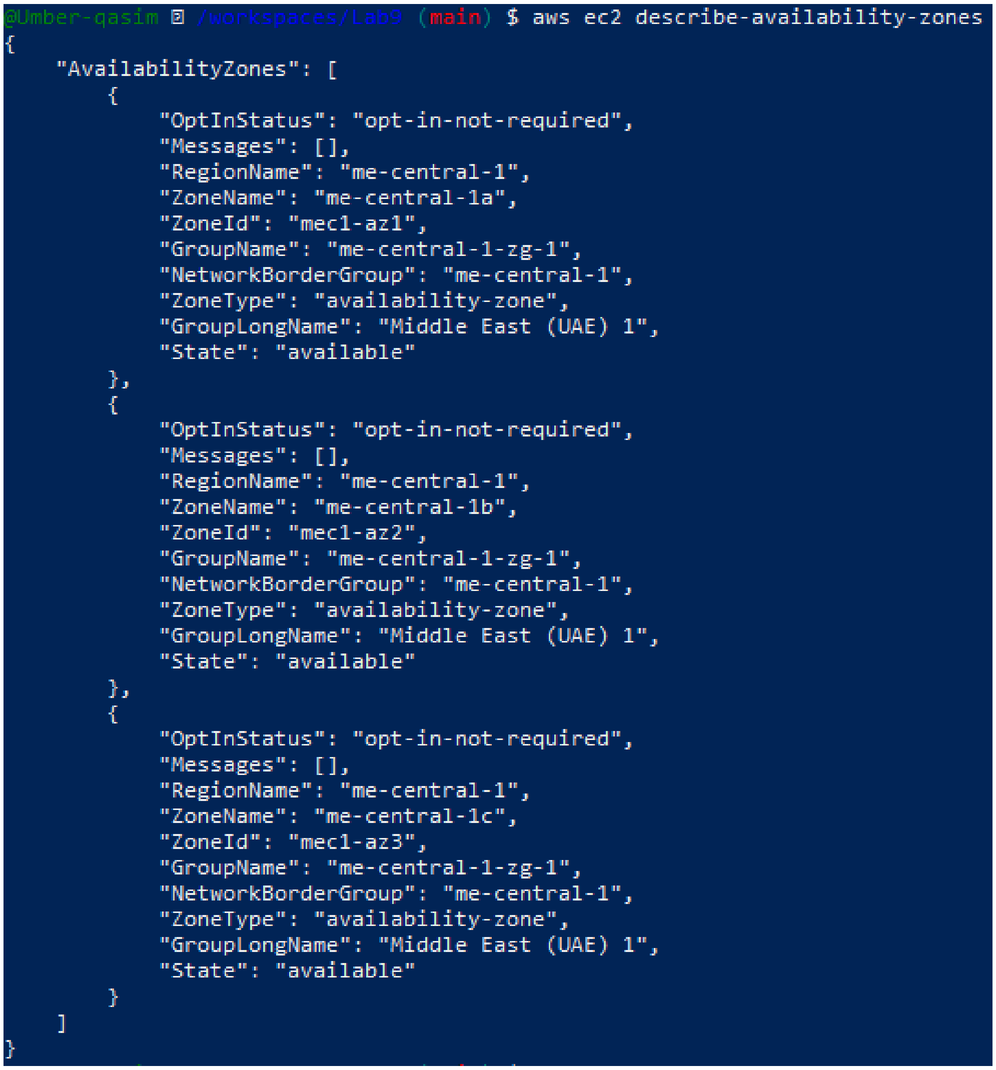
Regions



ec2-user@ip-172-31-20-197:~

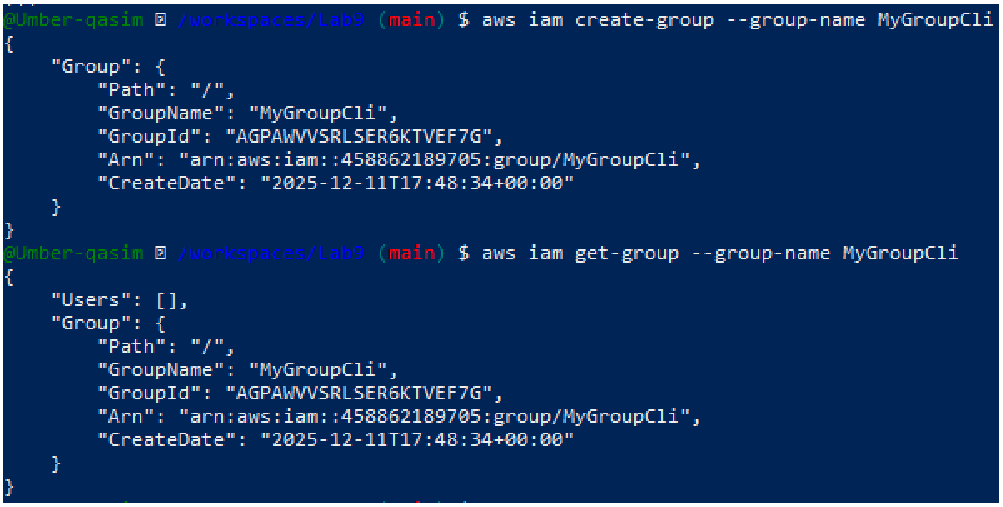
```
{
    "Regions": [
        {
            "OptInStatus": "opt-in-not-required",
            "RegionName": "ap-south-1",
            "Endpoint": "ec2.ap-south-1.amazonaws.com"
        },
        {
            "OptInStatus": "opt-in-not-required",
            "RegionName": "eu-north-1",
            "Endpoint": "ec2.eu-north-1.amazonaws.com"
        },
        {
            "OptInStatus": "opt-in-not-required",
            "RegionName": "eu-west-3",
            "Endpoint": "ec2.eu-west-3.amazonaws.com"
        },
        {
            "OptInStatus": "opt-in-not-required",
            "RegionName": "eu-west-2",
            "Endpoint": "ec2.eu-west-2.amazonaws.com"
        },
        {
            "OptInStatus": "opt-in-not-required",
            "RegionName": "eu-west-1",
            "Endpoint": "ec2.eu-west-1.amazonaws.com"
        },
        {
            "OptInStatus": "opt-in-not-required",
            "RegionName": "ap-northeast-3",
            "Endpoint": "ec2.ap-northeast-3.amazonaws.com"
        },
        {
            "OptInStatus": "opt-in-not-required",
            "RegionName": "ap-northeast-2",
            "Endpoint": "ec2.ap-northeast-2.amazonaws.com"
        },
        {
            "OptInStatus": "opt-in-not-required",
            "RegionName": "ap-northeast-1",
            "Endpoint": "ec2.ap-northeast-1.amazonaws.com"
        },
        {
            "OptInStatus": "opted-in",
            "RegionName": "me-central-1",
            "Endpoint": "ec2.me-central-1.amazonaws.com"
        },
        {
            "OptInStatus": "opt-in-not-required",
:
```

Availability Zones

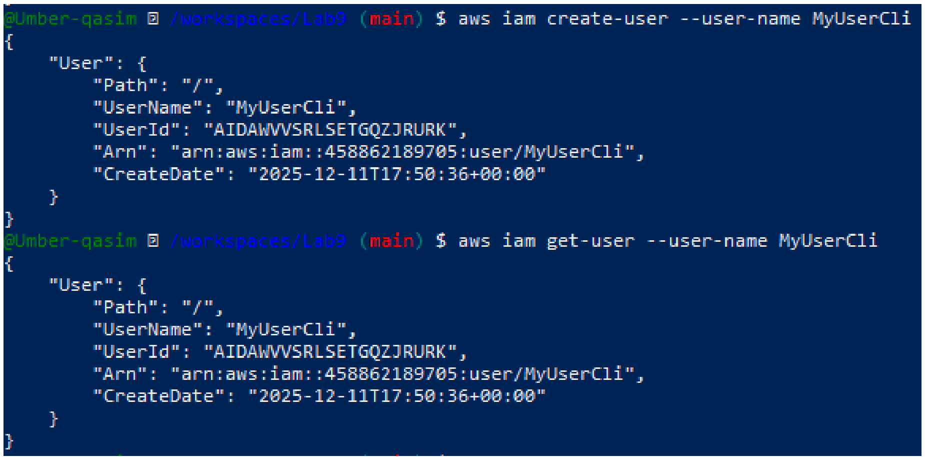


```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-availability-zones
{
    "AvailabilityZones": [
        {
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "me-central-1",
            "ZoneName": "me-central-1a",
            "ZoneId": "mec1-az1",
            "GroupName": "me-central-1-zg-1",
            "NetworkBorderGroup": "me-central-1",
            "ZoneType": "availability-zone",
            "GroupLongName": "Middle East (UAE) 1",
            "State": "available"
        },
        {
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "me-central-1",
            "ZoneName": "me-central-1b",
            "ZoneId": "mec1-az2",
            "GroupName": "me-central-1-zg-1",
            "NetworkBorderGroup": "me-central-1",
            "ZoneType": "availability-zone",
            "GroupLongName": "Middle East (UAE) 1",
            "State": "available"
        },
        {
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "me-central-1",
            "ZoneName": "me-central-1c",
            "ZoneId": "mec1-az3",
            "GroupName": "me-central-1-zg-1",
            "NetworkBorderGroup": "me-central-1",
            "ZoneType": "availability-zone",
            "GroupLongName": "Middle East (UAE) 1",
            "State": "available"
        }
    ]
}
```

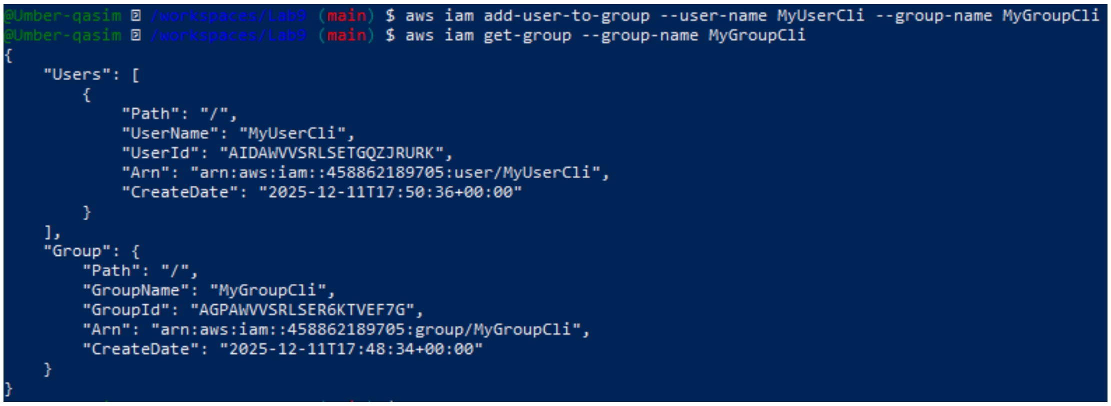
Task#06: IAM: create group, user, attach policies, create console login & keys



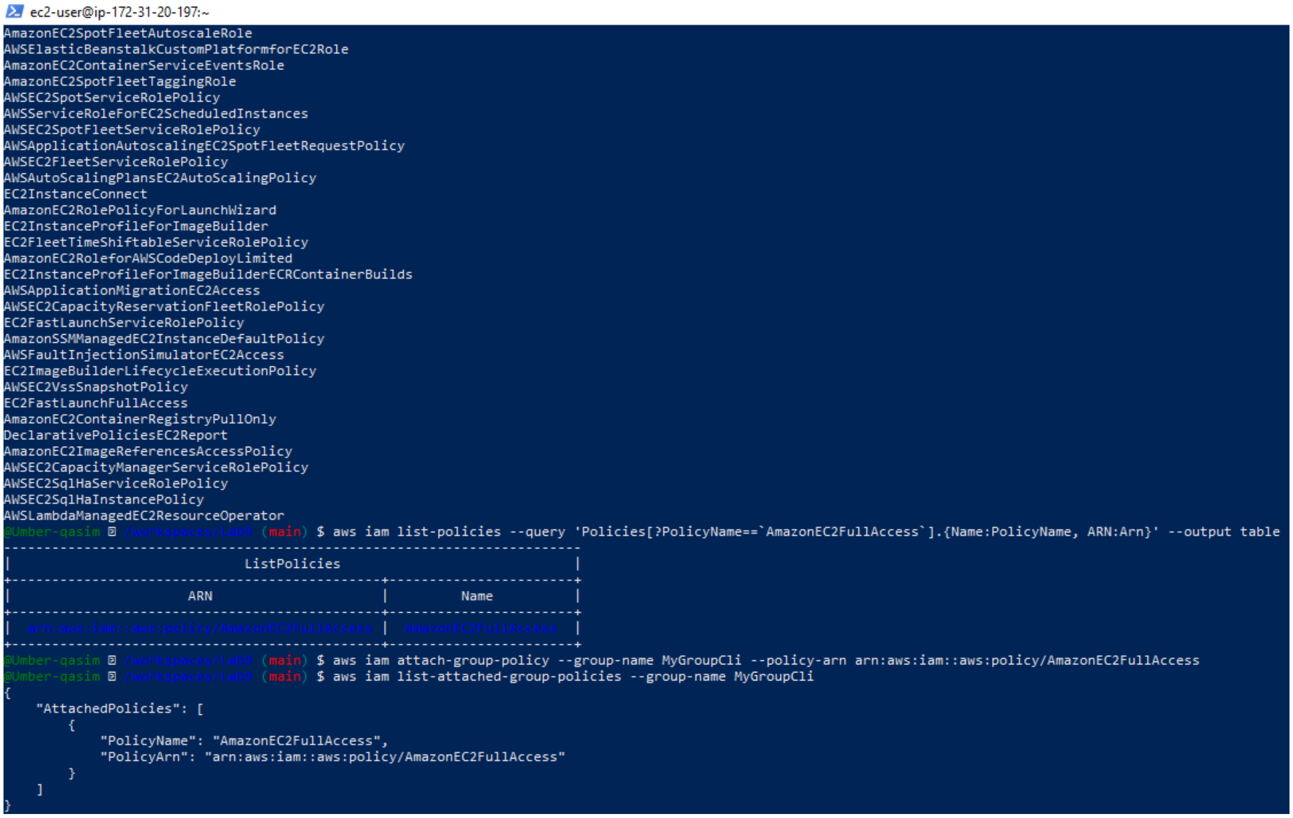
```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam create-group --group-name MyGroupCli
{
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAWVVSRLSER6KTVEF7G",
        "Arn": "arn:aws:iam::458862189705:group/MyGroupCli",
        "CreateDate": "2025-12-11T17:48:34+00:00"
    }
}
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam get-group --group-name MyGroupCli
{
    "Users": [],
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAWVVSRLSER6KTVEF7G",
        "Arn": "arn:aws:iam::458862189705:group/MyGroupCli",
        "CreateDate": "2025-12-11T17:48:34+00:00"
    }
}
```



```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam create-user --user-name MyUserCli
{
    "User": {
        "Path": "/",
        "UserName": "MyUserCli",
        "UserId": "AIDAWVVSRLSETGQZJRURK",
        "Arn": "arn:aws:iam::458862189705:user/MyUserCli",
        "CreateDate": "2025-12-11T17:50:36+00:00"
    }
}
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam get-user --user-name MyUserCli
{
    "User": {
        "Path": "/",
        "UserName": "MyUserCli",
        "UserId": "AIDAWVVSRLSETGQZJRURK",
        "Arn": "arn:aws:iam::458862189705:user/MyUserCli",
        "CreateDate": "2025-12-11T17:50:36+00:00"
    }
}
```

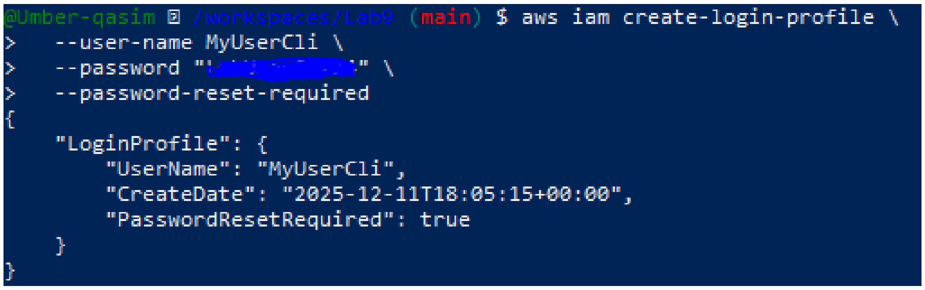


```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam add-user-to-group --user-name MyUserCli --group-name MyGroupCli
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam get-group --group-name MyGroupCli
{
    "Users": [
        {
            "Path": "/",
            "UserName": "MyUserCli",
            "UserId": "AIDAWVVSRLSETGQZJRURK",
            "Arn": "arn:aws:iam::458862189705:user/MyUserCli",
            "CreateDate": "2025-12-11T17:50:36+00:00"
        }
    ],
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAWVVSRLSER6KTVEF7G",
        "Arn": "arn:aws:iam::458862189705:group/MyGroupCli",
        "CreateDate": "2025-12-11T17:48:34+00:00"
    }
}
```

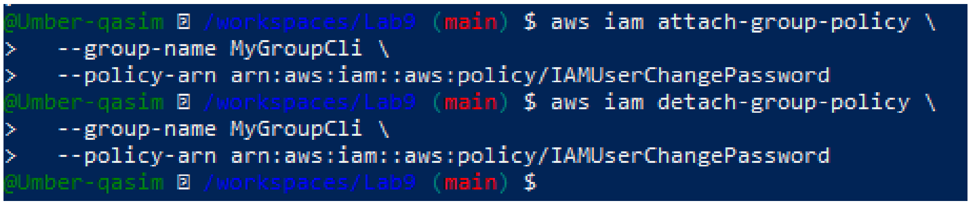


ec2-user@ip-172-31-20-197:~

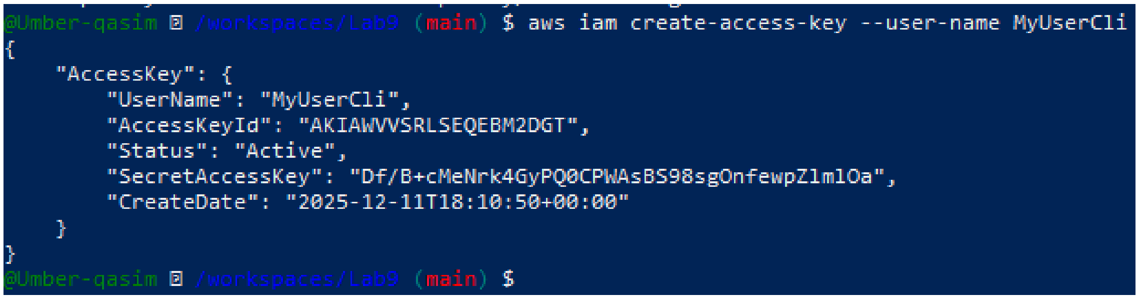
```
AmazonEC2SpotFleetAutoscaleRole
AWSElasticBeanstalkCustomPlatformforEC2Role
AmazonEC2ContainerServiceEventsRole
AmazonEC2SpotFleetTaggingRole
AWSEC2SpotServiceRolePolicy
AWSServiceRoleForEC2ScheduledInstances
AWSEC2SpotFleetServiceRolePolicy
AWSApplicationAutoscalingEC2SpotFleetRequestPolicy
AWSEC2FleetServiceRolePolicy
AWSAutoScalingPlansEC2AutoScalingPolicy
EC2InstanceConnect
AmazonEC2RolePolicyForLaunchWizard
EC2InstanceProfileForImageBuilder
EC2FleetTimeShiftableServiceRolePolicy
AmazonEC2RoleforAWSCodeDeployLimited
EC2InstanceProfileForImageBuilderECRContainerBuilds
AWSApplicationMigrationEC2Access
AWSEC2CapacityReservationFleetRolePolicy
EC2FastLaunchServiceRolePolicy
AmazonSSMManagedEC2InstanceDefaultPolicy
AWSFaultInjectionSimulatorEC2Access
EC2ImageBuilderLifecycleExecutionPolicy
AWSEC2VssSnapshotPolicy
EC2FastLaunchFullAccess
AmazonEC2ContainerRegistryPullOnly
DeclarativePoliciesEC2Report
AmazonEC2ImageReferencesAccessPolicy
AWSEC2CapacityManagerServiceRolePolicy
AWSEC2SqlHaServiceRolePolicy
AWSEC2SqlHaInstancePolicy
AWSLambdaManagedEC2ResourceOperator
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam list-policies --query 'Policies[?PolicyName==`AmazonEC2FullAccess`].{Name:PolicyName, ARN:Arn}' --output table
-------------------------------------------------------------------------
|                             ListPolicies                              |
+----------------------------------------------+------------------------+
|                     ARN                      |          Name          |
+----------------------------------------------+------------------------+
|  arn:aws:iam::aws:policy/AmazonEC2FullAccess |  AmazonEC2FullAccess   |
+----------------------------------------------+------------------------+
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam attach-group-policy --group-name MyGroupCli --policy-arn arn:aws:iam::aws:policy/AmazonEC2FullAccess
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam list-attached-group-policies --group-name MyGroupCli
{
    "AttachedPolicies": [
        {
            "PolicyName": "AmazonEC2FullAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"
        }
    ]
}
```



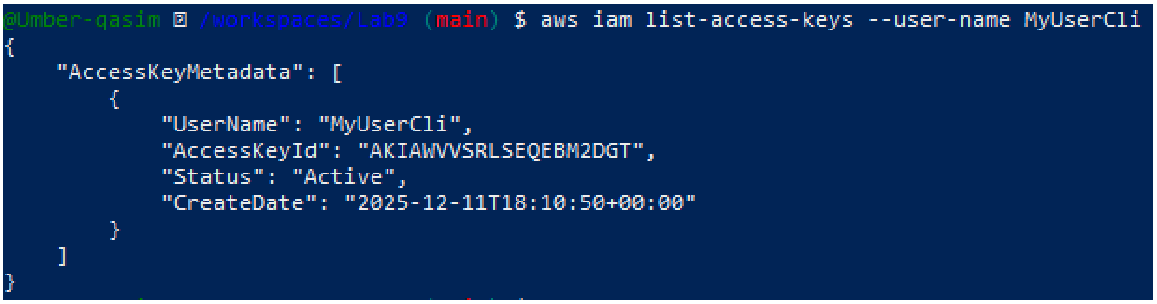
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam create-login-profile \
>    --user-name MyUserCli \
>    --password "[redacted]" \
>    --password-reset-required
{
    "LoginProfile": {
        "UserName": "MyUserCli",
        "CreateDate": "2025-12-11T18:05:15+00:00",
        "PasswordResetRequired": true
    }
}
```



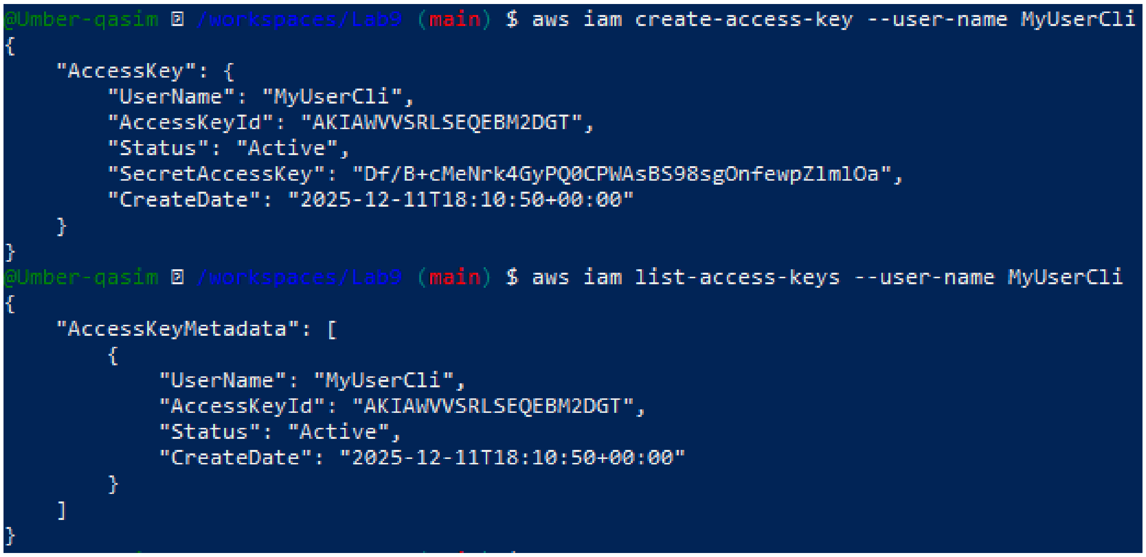
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam attach-group-policy \
>    --group-name MyGroupCli \
>    --policy-arn arn:aws:iam::aws:policy/IAMUserChangePassword
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam detach-group-policy \
>    --group-name MyGroupCli \
>    --policy-arn arn:aws:iam::aws:policy/IAMUserChangePassword
@Umber-qasim ⊡ /workspaces/Lab9 (main) $
```



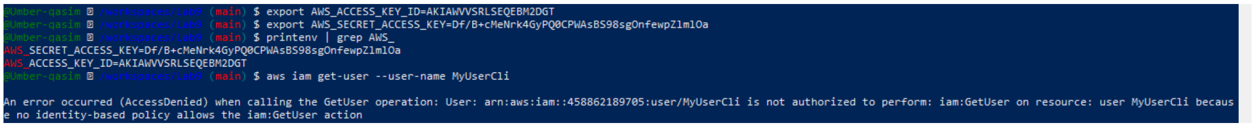
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam create-access-key --user-name MyUserCli
{
    "AccessKey": {
        "UserName": "MyUserCli",
        "AccessKeyId": "AKIAWVVSRLSEQEBM2DGT",
        "Status": "Active",
        "SecretAccessKey": "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",
        "CreateDate": "2025-12-11T18:10:50+00:00"
    }
}
@Umber-qasim ⊡ /workspaces/Lab9 (main) $
```



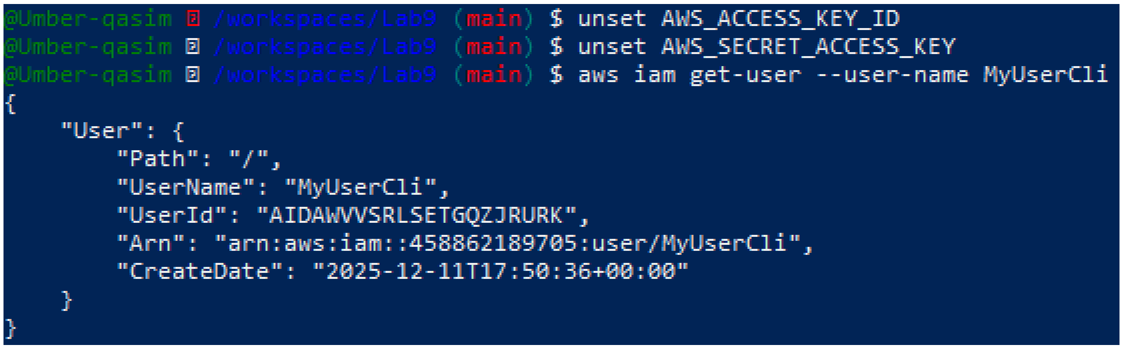
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam list-access-keys --user-name MyUserCli
{
    "AccessKeyMetadata": [
        {
            "UserName": "MyUserCli",
            "AccessKeyId": "AKIAWVVSRLSEQEBM2DGT",
            "Status": "Active",
            "CreateDate": "2025-12-11T18:10:50+00:00"
        }
    ]
}
```



```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam create-access-key --user-name MyUserCli
{
    "AccessKey": {
        "UserName": "MyUserCli",
        "AccessKeyId": "AKIAWVVSRLSEQEBM2DGT",
        "Status": "Active",
        "SecretAccessKey": "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",
        "CreateDate": "2025-12-11T18:10:50+00:00"
    }
}
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam list-access-keys --user-name MyUserCli
{
    "AccessKeyMetadata": [
        {
            "UserName": "MyUserCli",
            "AccessKeyId": "AKIAWVVSRLSEQEBM2DGT",
            "Status": "Active",
            "CreateDate": "2025-12-11T18:10:50+00:00"
        }
    ]
}
```

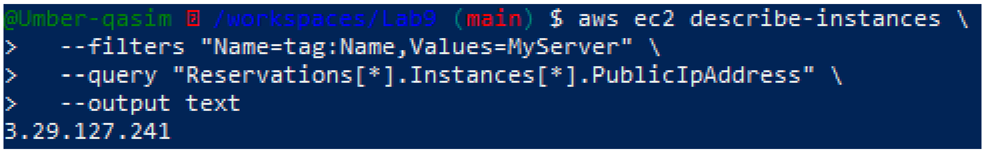


```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ export AWS_ACCESS_KEY_ID=AKIAWVVSRLSEQEBM2DGT
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ export AWS_SECRET_ACCESS_KEY=Df/B+cMeNrk4GyPQ0CPWAsBS98sgOnfewpZlml0a
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ printenv | grep AWS_
AWS_SECRET_ACCESS_KEY=Df/B+cMeNrk4GyPQ0CPWAsBS98sgOnfewpZlml0a
AWS_ACCESS_KEY_ID=AKIAWVVSRLSEQEBM2DGT
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam get-user --user-name MyUserCli

An error occurred (AccessDenied) when calling the GetUser operation: User: arn:aws:iam::458862189705:user/MyUserCli is not authorized to perform: iam:GetUser on resource: user MyUserCli becaus
e no identity-based policy allows the iam:GetUser action
```

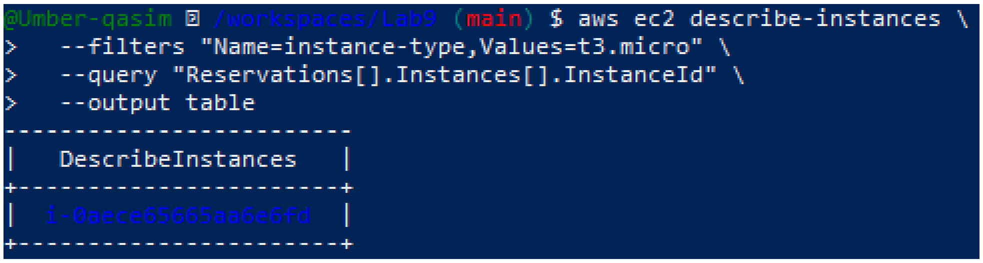


```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ unset AWS_ACCESS_KEY_ID
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ unset AWS_SECRET_ACCESS_KEY
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws iam get-user --user-name MyUserCli
{
    "User": {
        "Path": "/",
        "UserName": "MyUserCli",
        "UserId": "AIDAWVVSRLSETGQZJRURK",
        "Arn": "arn:aws:iam::458862189705:user/MyUserCli",
        "CreateDate": "2025-12-11T17:50:36+00:00"
    }
}
```

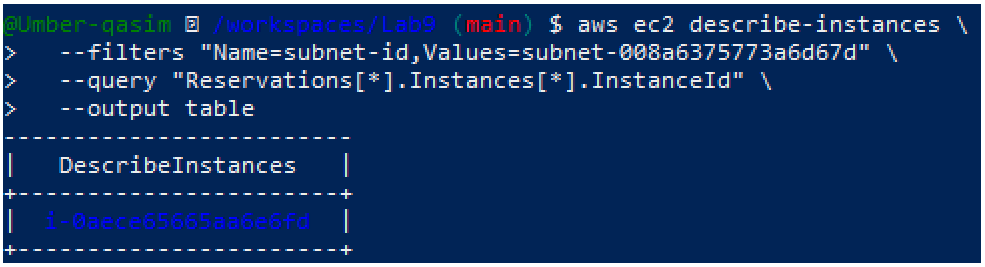
Task#07: Filters: query with filters to find instances and their attributes



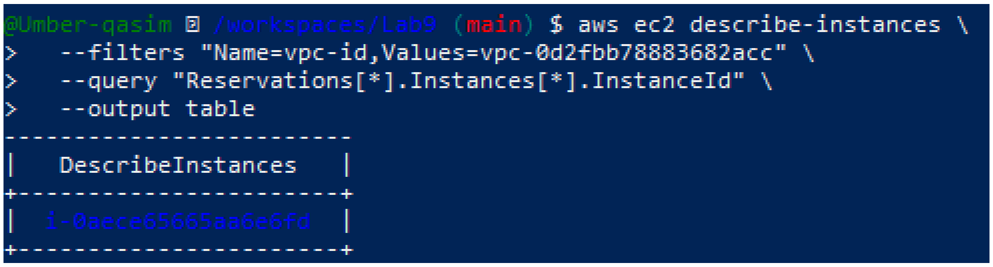
```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-instances \
>   --filters "Name=tag:Name,Values=MyServer" \
>   --query "Reservations[*].Instances[*].PublicIpAddress" \
>   --output text
3.29.127.241
```



```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-instances \
>   --filters "Name=instance-type,Values=t3.micro" \
>   --query "Reservations[].Instances[].InstanceId" \
>   --output table
-------------------------
|   DescribeInstances   |
+-----------------------+
|  i-0aece65665aa6e6fd  |
+-----------------------+
```

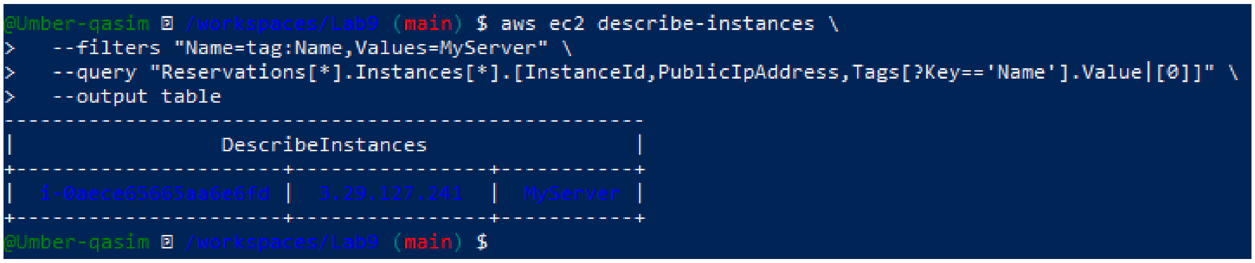


```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-instances \
>   --filters "Name=subnet-id,Values=subnet-008a6375773a6d67d" \
>   --query "Reservations[*].Instances[*].InstanceId" \
>   --output table
-------------------------
|   DescribeInstances   |
+-----------------------+
|  i-0aece65665aa6e6fd  |
+-----------------------+
```

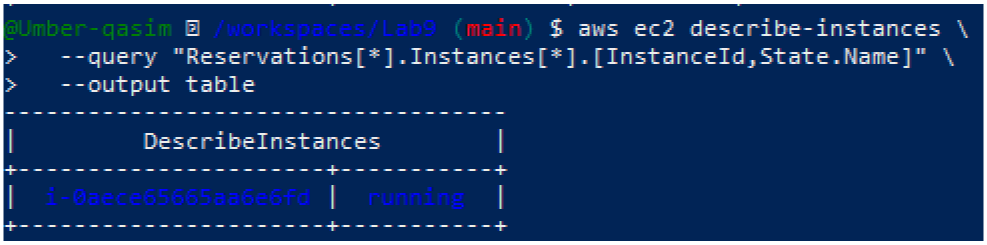


```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-instances \
>   --filters "Name=vpc-id,Values=vpc-0d2fbb78883682acc" \
>   --query "Reservations[*].Instances[*].InstanceId" \
>   --output table
-------------------------
|   DescribeInstances   |
+-----------------------+
|  i-0aece65665aa6e6fd  |
+-----------------------+
```

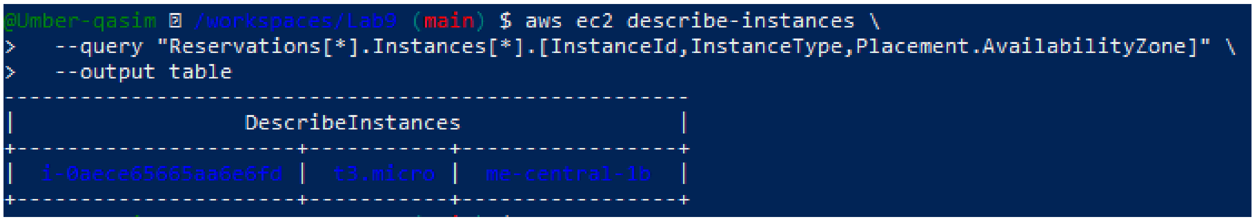
Task#08: Use --query to format outputs for reporting



```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-instances \
>   --filters "Name=tag:Name,Values=MyServer" \
>   --query "Reservations[*].Instances[*].[InstanceId,PublicIpAddress,Tags[?Key=='Name'].Value|[0]]" \
>   --output table
-----------------------------------------------------
|                 DescribeInstances                 |
+----------------------+----------------+-----------+
|  i-0aece65665aa6e6fd |  3.29.127.241  |  MyServer |
+----------------------+----------------+-----------+
@Umber-qasim ⍰ /workspaces/Lab9 (main) $
```

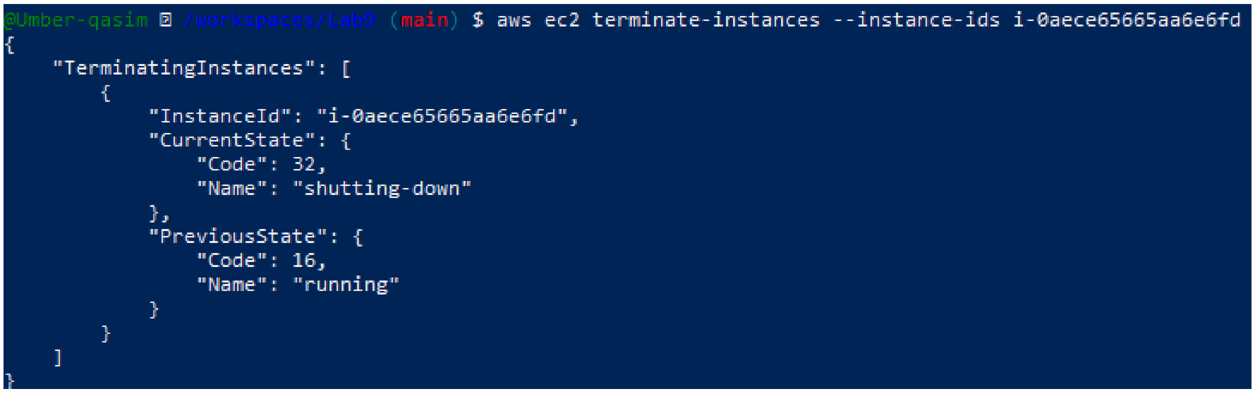


```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-instances \
>   --query "Reservations[*].Instances[*].[InstanceId,State.Name]" \
>   --output table
------------------------------------
|         DescribeInstances        |
+----------------------+-----------+
|  i-0aece65665aa6e6fd |  running  |
+----------------------+-----------+
```



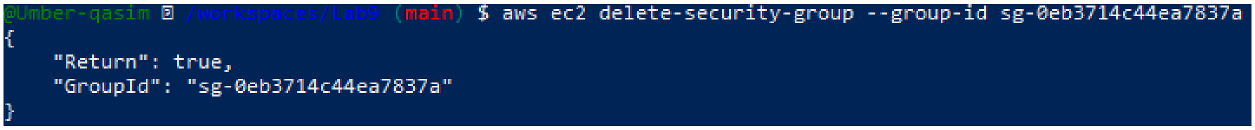
```
@Umber-qasim ⍰ /workspaces/Lab9 (main) $ aws ec2 describe-instances \
>   --query "Reservations[*].Instances[*].[InstanceId,InstanceType,Placement.AvailabilityZone]" \
>   --output table
---------------------------------------------------------
|                   DescribeInstances                   |
+----------------------+------------+-------------------+
|  i-0aece65665aa6e6fd |  t3.micro  |  me-central-1b    |
+----------------------+------------+-------------------+
```

Terminal EC2 Instance



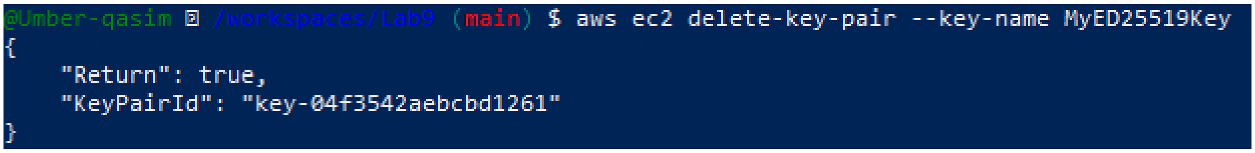
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws ec2 terminate-instances --instance-ids i-0aece65665aa6e6fd
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-0aece65665aa6e6fd",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

Delete Security Group



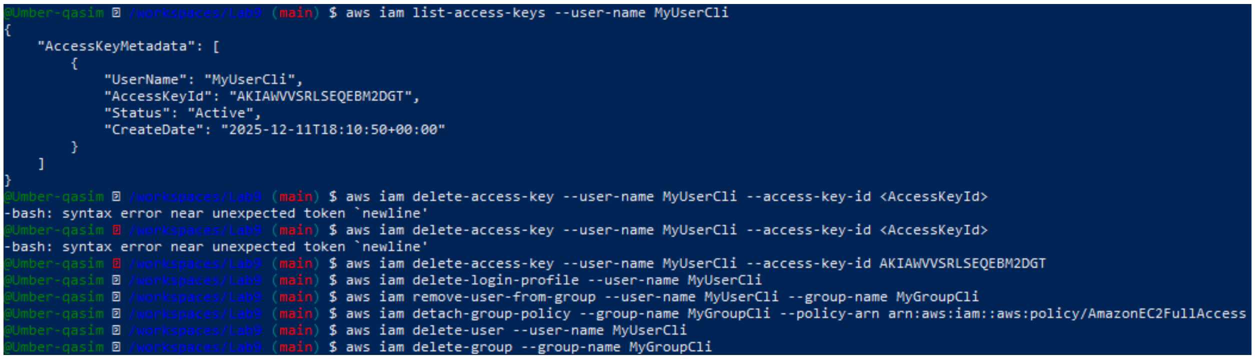
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws ec2 delete-security-group --group-id sg-0eb3714c44ea7837a
{
    "Return": true,
    "GroupId": "sg-0eb3714c44ea7837a"
}
```

Delete Key Pair



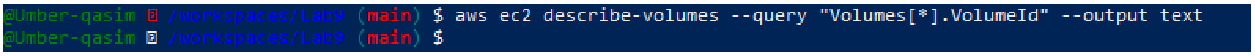
```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws ec2 delete-key-pair --key-name MyED25519Key
{
    "Return": true,
    "KeyPairId": "key-04f3542aebcbd1261"
}
```

Delete IAM User’s Access Key, Delete IAM Login Profile, Remove User from Group, Detach Policy from Group, Delete IAM User, Delete IAM Group



```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam list-access-keys --user-name MyUserCli
{
    "AccessKeyMetadata": [
        {
            "UserName": "MyUserCli",
            "AccessKeyId": "AKIAWVVSRLSEQEBM2DGT",
            "Status": "Active",
            "CreateDate": "2025-12-11T18:10:50+00:00"
        }
    ]
}
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam delete-access-key --user-name MyUserCli --access-key-id <AccessKeyId>
-bash: syntax error near unexpected token `newline'
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam delete-access-key --user-name MyUserCli --access-key-id <AccessKeyId>
-bash: syntax error near unexpected token `newline'
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam delete-access-key --user-name MyUserCli --access-key-id AKIAWVVSRLSEQEBM2DGT
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam delete-login-profile --user-name MyUserCli
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam remove-user-from-group --user-name MyUserCli --group-name MyGroupCli
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam detach-group-policy --group-name MyGroupCli --policy-arn arn:aws:iam::aws:policy/AmazonEC2FullAccess
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam delete-user --user-name MyUserCli
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws iam delete-group --group-name MyGroupCli
```

Optional: Delete any leftover EBS volumes/snapshots



```
@Umber-qasim ⊡ /workspaces/Lab9 (main) $ aws ec2 describe-volumes --query "Volumes[*].VolumeId" --output text
@Umber-qasim ⊡ /workspaces/Lab9 (main) $
```