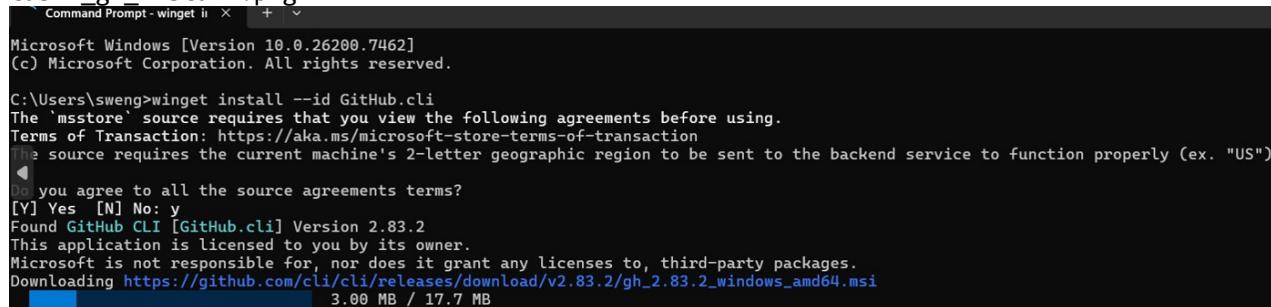


## CLOUD COMPUTING LAB

**Name: Nayab Khazin Reg No: 2023-BSE-046**  
**Submitted to: Sir Muhammad Shoaib Class: 5B**  
**LAB #09**

### Task 1 — GitHub CLI, Codespace setup and authentication

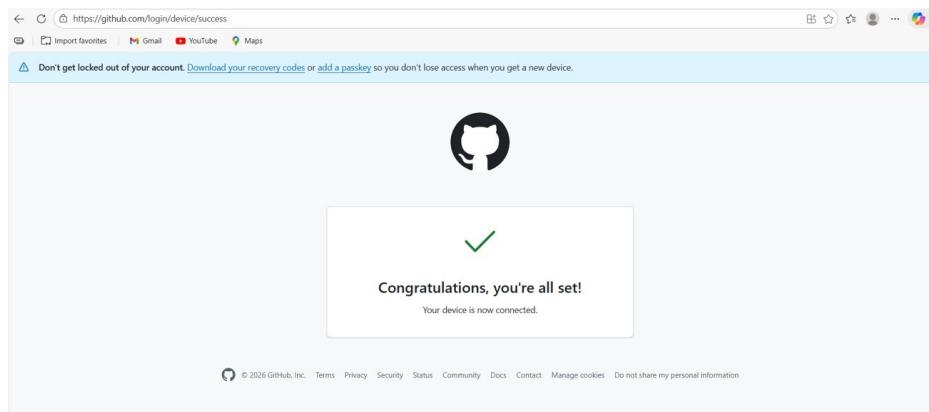
task1\_gh\_install.png



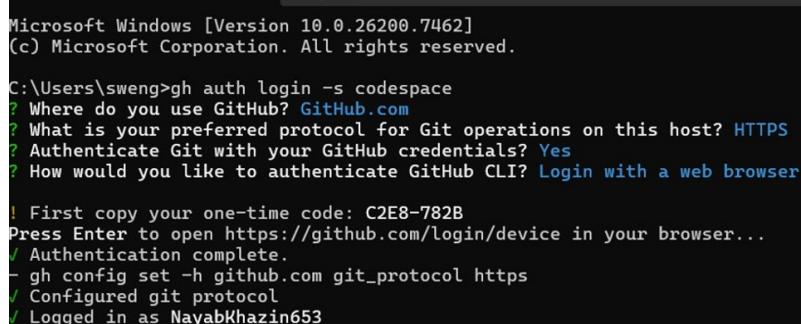
```
Microsoft Windows [Version 10.0.26200.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sweng>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 properly (ex. "US")
Do you agree to all the source agreements terms?
[Y] Yes [N] No: y
Found GitHub CLI [GitHub.cli] Version 2.83.2
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/v2.83.2/gh_2.83.2_windows_amd64.msi
    3.00 MB / 17.7 MB
```

task1\_gh\_auth\_login.png



task1\_gh\_auth\_login\_command.png



```
Microsoft Windows [Version 10.0.26200.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sweng>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? Login with a web browser

! First copy your one-time code: C2E8-782B
Press Enter to open https://github.com/login/device in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as NayabKhazin653
```

task1\_codespace\_list.png

```
C:\Users\sweng>gh codespace list
NAME          DISPLAY NAME    REPOSITORY      BRANCH STATE    CREATED AT
literate-winner-x5g5wr75vw4jcpv7j  literate winner  NayabKhazin653/LAB-9  main Available about 2 minutes ago
C:\Users\sweng>
```

task1\_codespace\_ssh\_connected.png

```
C:\Users\sweng>gh codespace ssh -c literate-winner-x5g5wr75vw4jcpv7j
Enter passphrase for key 'C:\Users\sweng/.ssh/id_ed25519':
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.

@NavabKhazin653 → /workspaces/LAB-9 (main) $ |
```

## Task 2 — Install AWS CLI inside the Codespace and configure it

task2\_aws\_install.png

```
inflating: aws/dist/awscli/topics/s3-faq.rst
inflating: aws/dist/awscli/topics/topic-tags.json
inflating: aws/dist/awscli/topics/ddb-expressions.rst
inflating: aws/dist/awscli/topics/s3-config.rst
inflating: aws/dist/awscli/topics/return-codes.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/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/wizard/wizards/iam/new-role.yml
inflating: aws/dist/awscli/customizations/sso/index.html
inflating: aws/dist/awscli/data/metadata.json
inflating: aws/dist/awscli/data/cli.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/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/INSTALLER
inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/RECORD
inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/licenses/AUTHORS.rst
inflating: aws/dist/prompt_toolkit-3.0.51.dist-info/licenses/LICENSE
inflating: aws/dist/wheel-0.45.1.dist-info/WHEEL
inflating: aws/dist/wheel-0.45.1.dist-info/LICENSE.txt
inflating: aws/dist/wheel-0.45.1.dist-info/METADATA
inflating: aws/dist/wheel-0.45.1.dist-info/REQUESTED
inflating: aws/dist/wheel-0.45.1.dist-info/INSTALLER
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
@NayabKhazin653 → /workspaces/LAB-9 (main) $ sudo ./aws/install
You can now run: /usr/local/bin/aws --version
@NayabKhazin653 → /workspaces/LAB-9 (main) $
```

task2\_aws\_install\_and\_version.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws --version
aws-cli/2.32.28 Python/3.13.11 Linux/6.8.0-1030-azure exe/x86_64.ubuntu.24
@NayabKhazin653 → /workspaces/LAB-9 (main) $ |
```

task2\_aws\_configure\_and\_files.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws configure
AWS Access Key ID [None]: AKIAUHXY42RXNJYIXKD6
AWS Secret Access Key [None]: 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
Default region name [None]: me-central-1
Default output format [None]: json
```

task2\_aws\_get\_caller\_identity.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws sts get-caller-identity
{
  "UserId": "AIDAUHXY42RXF3IJLIPXL",
  "Account": "291506017390",
  "Arn": "arn:aws:iam::291506017390:user/Admin"
}
```

Task 3 – Create security group and add ingress rules using Codespace IP:

task3\_create\_security\_group\_output.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 create-security-group --group-name 'MySecurityGroup' \
>   --description 'My Security Group' \
>   --vpc-id 'vpc-0e3a7ba238e097490'
{
  "GroupId": "sg-006be544db1a757c8",
  "SecurityGroupArn": "arn:aws:ec2:me-central-1:291506017390:security-group/sg-006be544db1a757c8"
}
```

task3\_describe\_sg\_before\_ingress.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-security-groups --group-ids sg-006be544db1a757c8
{
  "SecurityGroups": [
    {
      "GroupId": "sg-006be544db1a757c8",
      "IpPermissionsEgress": [
        {
          "IpProtocol": "-1",
          "UserIdGroupPairs": [],
          "IpRanges": [
            {
              "CidrIp": "0.0.0.0/0"
            }
          ],
          "Ipv6Ranges": [],
          "PrefixListIds": []
        }
      ],
      "VpcId": "vpc-0e3a7ba238e007490",
      "SecurityGroupArn": "arn:aws:ec2:me-central-1:291506017390:security-group/sg-006be544db1a757c8",
      "OwnerId": "291506017390",
      "GroupName": "MySecurityGroup",
      "Description": "My Security Group",
      "IpPermissions": []
    }
  ]
}
```

task3\_codespace\_public\_ip.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ curl icanhazip.com
13.71.3.97
```

task3\_authorize\_ssh.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 authorize-security-group-ingress \
> --group-id sg-006be544db1a757c8 \
> --protocol tcp \
> --port 22 \
> --cidr 13.71.3.97/32
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-0e805ff79baa67f38",
      "GroupId": "sg-006be544db1a757c8",
      "GroupOwnerId": "291506017390",
      "IsEgress": false,
      "IpProtocol": "tcp",
      "FromPort": 22,
      "ToPort": 22,
      "CidrIpv4": "13.71.3.97/32",
      "SecurityGroupRuleArn": "arn:aws:ec2:me-central-1:291506017390:security-group-rule/sgr-0e805ff79baa67f38"
    }
  ]
}
```

task3\_describe.png

```
NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-security-groups --group-ids sg-006be544db1a757c8
{
    "SecurityGroups": [
        {
            "GroupId": "sg-006be544db1a757c8",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-0e3a7ba238e097490",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:291506017390:security-group/sg-006be544db1a757c8",
            "OwnerId": "291506017390",
            "GroupName": "MySecurityGroup",
            "Description": "My Security Group",
            "IpPermissions": [
                {
                    "IpProtocol": "tcp",
                    "FromPort": 22,
                    "ToPort": 22,
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "13.71.3.97/32"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ]
        }
    ]
}
```

#### task3\_authorize\_http\_and\_describe.png

```
NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 authorize-security-group-ingress \
> --group-id sg-006be544db1a757c8 \
> --ip-permissions '{"FromPort":80,"ToPort":80,"IpProtocol":"tcp","IpRanges":[{"CidrIp":"13.71.3.97/32"}]}'
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-061271e523ce54c97",
            "GroupId": "sg-006be544db1a757c8",
            "GroupOwnerId": "291506017390",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "13.71.3.97/32",
            "SecurityGroupRuleArn": "arn:aws:ec2:me-central-1:291506017390:security-group-rule/sgr-061271e523ce54c97"
        }
    ]
}
```

#### task3\_describe\_sg\_final.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-security-groups --group-ids sg-006be544db1a757c8
{
    "SecurityGroups": [
        {
            "GroupId": "sg-006be544db1a757c8",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-0e3a7ba238e097490",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:291506017390:security-group/sg-006be544db1a757c8",
            "OwnerId": "291506017390",
            "GroupName": "MySecurityGroup",
            "Description": "My Security Group",
            "IpPermissions": [
                {
                    "IpProtocol": "tcp",
                    "FromPort": 80,
                    "ToPort": 80,
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "13.71.3.97/32"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                },
                {
                    "IpProtocol": "tcp",
                    "FromPort": 22,
                    "ToPort": 22,
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "13.71.3.97/32"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ]
        }
    ]
}
```

## Task 4 — Create a key pair, describe key pairs, and launch EC2 instance

task4\_create\_keypair\_output.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 create-key-pair \
>   --key-name MyED25519Key \
>   --key-type ed25519 \
>   --key-format pem \
>   --query 'KeyMaterial' \
>   --output text > MyED25519Key.pem
```

task4\_describe\_keypairs.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-key-pairs
{
  "KeyPairs": [
    {
      "KeyPairId": "key-0b3321e4b62c4b840",
      "KeyType": "ed25519",
      "Tags": [],
      "CreateTime": "2026-01-06T10:01:58.286000+00:00",
      "KeyName": "MyED25519Key",
      "KeyFingerprint": "hrChoNMA65kihbam0ALAncj7DjBaJ3R5lFSuSiHXiYs="
    }
  ]
}
```

#### task4\_run\_instances\_output.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 run-instances \
--image-id ami-05e66df2bafcb7dea \
--count 1 \
--instance-type t3.micro \
--key-name MyED25519Key \
--security-group-ids sg-006be544db1a757c8 \
--subnet-id subnet-0e02d581cf0ee452 \
--tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=MyServer}]"
{
  "ReservationId": "r-0fac0d6675ffaf878",
  "OwnerId": "291506017390",
  "Groups": [],
  "Instances": [
    {
      "Architecture": "x86_64",
      "BlockDeviceMappings": [],
      "ClientToken": "372f1e5c-c8cc-4a4a-bec5-5f9995cf44e1",
      "EbsOptimized": false,
      "EnaSupport": true,
      "Hypervisor": "xen",
      "NetworkInterfaces": [
        {
          "Attachment": {
            "AttachTime": "2026-01-06T13:14:00+00:00",
            "AttachmentId": "eni-attach-03795d98af75436f7",
            "DeleteOnTermination": true,
            "DeviceIndex": 0,
            "Status": "attaching",
            "NetworkCardIndex": 0
          },
          "Description": "",
          "Groups": [
            {
              "GroupId": "sg-006be544db1a757c8",
              "GroupName": "MySecurityGroup"
            }
          ],
          "Ipv6Addresses": [],
          "MacAddress": "0a:ad:95:4c:15:f5",
          "NetworkInterfaceId": "eni-0fd644f08a9f5dbcb",
          "OwnerId": "291506017390",
          "PrivateDnsName": "ip-172-31-24-116.me-central-1.compute.internal",
          "PrivateIpAddress": "172.31.24.116",
        }
      ]
    }
  ]
}
```

#### task4\_describe\_instances\_public\_ip.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-instances --query "Reservations[*].Inst
PublicIpAddress,State.Name" --output table
-----|-----|-----|
| DescribeInstances |-----|-----|
| i-0fdbb82cc7e99be09 | 51.112.142.8 | running |
|-----|-----|-----|
```

## Task 5 – Understand AWS describe-\* commands

### task5\_describe\_security\_groups.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-security-groups
{
  "SecurityGroups": [
    {
      "GroupId": "sg-006be544db1a757c8",
      "IpPermissionsEgress": [
        {
          "IpProtocol": "-1",
          "UserIdGroupPairs": [],
          "IpRanges": [
            {
              "CidrIp": "0.0.0.0/0"
            }
          ],
          "Ipv6Ranges": [],
          "PrefixListIds": []
        }
      ],
      "VpcId": "vpc-0e3a7ba238e097490",
      "SecurityGroupArn": "arn:aws:ec2:me-central-1:291506017390:security-group/sg-006be544db1a757c8",
      "OwnerId": "291506017390",
      "GroupName": "MySecurityGroup",
      "Description": "My Security Group",
      "IpPermissions": [
        {
          "IpProtocol": "tcp",
          "FromPort": 80,
          "ToPort": 80,
          "UserIdGroupPairs": [],
          "IpRanges": [
            ...
          ]
        }
      ],
      "SecurityGroups": [
        {
          "GroupId": "sg-006be544db1a757c8",
          "IpPermissionsEgress": [
            {
              "IpProtocol": "-1",
              ...
            }
          ]
        }
      ]
    }
  ]
}
```

### task5\_describe\_vpcs.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-vpcs
{
  "Vpcs": [
    {
      "OwnerId": "291506017390",
      "InstanceTenancy": "default",
      "CidrBlockAssociationSet": [
        {
          "AssociationId": "vpc-cidr-assoc-0decd793abda69466",
          "CidrBlock": "172.31.0.0/16",
          "CidrBlockState": {
            "State": "associated"
          }
        }
      ],
      "IsDefault": true,
      "BlockPublicAccessStates": {
        "InternetGatewayBlockMode": "off"
      },
      "VpcId": "vpc-0e3a7ba238e097490",
      "State": "available",
      "CidrBlock": "172.31.0.0/16",
      "DhcpOptionsId": "dopt-09d2afde96c48ccf8"
    }
  ]
}
```

### task5\_describe\_subnets.png

```
NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-subnets
{
  "Subnets": [
    {
      "AvailabilityZoneId": "mec1-az1",
      "MapCustomerOwnedIpOnLaunch": false,
      "OwnerId": "291506017390",
      "AssignIpv6AddressOnCreation": false,
      "Ipv6CidrBlockAssociationSet": [],
      "SubnetArn": "arn:aws:ec2:me-central-1:291506017390:subnet/subnet-06a59b3771",
      "EnableDns64": false,
      "Ipv6Native": false,
      "PrivateDnsNameOptionsOnLaunch": {
        "HostnameType": "ip-name",
        "EnableResourceNameDnsARecord": false,
        "EnableResourceNameDnsAAAARecord": false
      },
      "BlockPublicAccessStates": {
        "InternetGatewayBlockMode": "off"
      },
      "SubnetId": "subnet-06a59b3771518f03a",
      "State": "available",
      "VpcId": "vpc-0e3a7ba238e097490",
      "CidrBlock": "172.31.32.0/20",
      "AvailableIpAddressCount": 4091,
      "AvailabilityZone": "me-central-1a",
      "DefaultForAz": true,
      "MapPublicIpOnLaunch": true
    },
    {
      "AvailabilityZoneId": "mec1-az2",
      "MapCustomerOwnedIpOnLaunch": false,
      "OwnerId": "291506017390",
      "AssignIpv6AddressOnCreation": false,
      "Ipv6CidrBlockAssociationSet": [],
      "SubnetArn": "arn:aws:ec2:me-central-1:291506017390:subnet/subnet-0e02d581cf",
      "EnableDns64": false,
      "Ipv6Native": false,
    }
  ]
}
```

task5\_describe\_instances.png

```
NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-instances
{
  "Reservations": [
    {
      "ReservationId": "r-0fac0d6675ffaf878",
      "OwnerId": "291506017390",
      "Groups": [],
      "Instances": [
        {
          "Architecture": "x86_64",
          "BlockDeviceMappings": [
            {
              "DeviceName": "/dev/xvda",
              "Ebs": {
                "AttachTime": "2026-01-06T13:14:01+00:00",
                "DeleteOnTermination": true,
                "Status": "attached",
                "VolumeId": "vol-0fa829739547640a6"
              }
            }
          ],
          "ClientToken": "372f1e5c-c8cc-4a4a-bec5-5f9995cf44e1",
          "EbsOptimized": false,
          "EnaSupport": true,
          "Hypervisor": "xen",
          "NetworkInterfaces": [
            {
              "Association": {
                "IpOwnerId": "amazon",
                "PublicDnsName": "ec2-51-112-142-8.me-central-1.com",
                "PublicIp": "51.112.142.8"
              },
              "Attachment": {
                "AttachTime": "2026-01-06T13:14:00+00:00",
                "AttachmentId": "eni-attach-03795d98af75436f7",
                "DeleteOnTermination": true,
                "DeviceIndex": 0,
                "Status": "attached",
                "NetworkCardIndex": 0
              },
              "Description": "",
              "Groups": []
            }
          ]
        }
      ]
    }
  ]
}
```

task5\_describe\_regions.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-regions

"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"
    }
],
```

task5\_describe\_availability\_zones.png

```
@NayabKhazin653 → /workspaces/LAB-9 (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"
        }
    ]
}
```

task6\_create\_group\_and\_user.png

## Task 6 — IAM: create group, user, attach policies, create console login & keys

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws iam create-group --group-name MyGroupCli
{
  "Group": {
    "Path": "/",
    "GroupName": "MyGroupCli",
    "GroupId": "AGPAUHXY42RXNQM5VYWL",
    "Arn": "arn:aws:iam::291506017390:group/MyGroupCli",
    "CreateDate": "2026-01-06T14:18:06+00:00"
  }
}
```

task6\_create\_group\_and\_user.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws iam get-group --group-name MyGroupCli
{
  "Users": [],
  "Group": {
    "Path": "/",
    "GroupName": "MyGroupCli",
    "GroupId": "AGPAUHXY42RXNQM5VYWL",
    "Arn": "arn:aws:iam::291506017390:group/MyGroupCli",
    "CreateDate": "2026-01-06T14:18:06+00:00"
  }
}
```

task6\_create\_group\_and\_user.png

task6\_create\_group\_and\_user.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws iam get-user --user-name MyUserCli
{
  "User": {
    "Path": "/",
    "UserName": "MyUserCli",
    "UserId": "AIDAUHXY42RXNXZ4ISGJZ",
    "Arn": "arn:aws:iam::291506017390:user/MyUserCli",
    "CreateDate": "2026-01-06T14:19:06+00:00"
  }
}
```

task6\_add\_user\_to\_group\_and\_verify.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws iam add-user-to-group --user-name MyUserCli --group-name MyGroupCli
```

task6\_add\_user\_to\_group\_and\_verify.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws iam get-group --group-name MyGroupCli
{
    "Users": [
        {
            "Path": "/",
            "UserName": "MyUserCli",
            "UserId": "AIDAUHXY42RXWZ41SGJZ",
            "Arn": "arn:aws:iam::291506017390:user/MyUserCli",
            "CreateDate": "2026-01-06T14:19:06+00:00"
        }
    ],
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAUHXY42RXWZ41SGJZ",
        "Arn": "arn:aws:iam::291506017390:group/MyGroupCli",
        "CreateDate": "2026-01-06T14:18:06+00:00"
    }
}
```

task6\_policy\_list\_and\_attach.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws iam list-policies \
> --query "Policies[?contains(PolicyName, 'EC2')].{Name:PolicyName}" \
> --output text
AmazonEC2FullAccess
AmazonEC2ReadOnlyAccess
AmazonElasticMapReduceForEC2Role
AmazonEC2RoleforDataPipelineRole
AmazonEC2ContainerServiceForEC2Role
AmazonEC2ContainerServiceRole
AmazonEC2RoleforAWSCodeDeploy
AmazonEC2RoleforSSM
CloudWatchActionsEC2Access
AmazonEC2ContainerRegistryReadOnly
AmazonEC2ContainerRegistryPowerUser
AmazonEC2ContainerRegistryFullAccess
AmazonEC2ContainerServiceAutoscaleRole
AmazonEC2SpotFleetAutoscaleRole
AWSelasticBeanstalkCustomPlatformForEC2Role
AmazonEC2ContainerServiceEventsRole
AmazonEC2SpotFleetTaggingRole
AWSSEC2SpotServiceRolePolicy
AWSServiceRoleForEC2ScheduledInstances
AWSSEC2SpotFleetServiceRolePolicy
AWSApplicationAutoscalingEC2SpotFleetRequestPolicy
AWSSEC2FleetServiceRolePolicy
AWSAutoScalingPlansEC2AutoScalingPolicy
EC2InstanceConnect
AmazonEC2RolePolicyForLaunchWizard
EC2InstanceProfileForImageBuilder
EC2FleetTimeShiftableServiceRolePolicy
AmazonEC2RoleforAWSCodeDeployLimited
EC2InstanceProfileForImageBuilderForContainerBuilds
AWSApplicationMigrationEC2Access
AWSSEC2CapacityReservationFleetRolePolicy
EC2FastLaunchServiceRolePolicy
AmazonSSMManagedEC2InstanceDefaultPolicy
AWSFaultInjectionSimulatorEC2Access
EC2ImageBuilderLifecycleExecutionPolicy
AWSSEC2VssSnapshotPolicy
EC2FastLaunchFullAccess
```

task6\_policy\_list\_and\_attach.png

```
@NayabKhazin653 → /workspaces/LAB-9 (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 |
+-----+-----+
```

task6\_policy\_list\_and\_attach.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws iam attach-group-policy \
>   --group-name MyGroupCli \
>   --policy-arn arn:aws:iam::aws:policy/AmazonEC2FullAccess
```

task6\_policy\_list\_and\_attach.png

```
NayabKhzain653 eworkspaces/Lab-9 (main) $ aws iam list-attached-group-policies --group-name MyGroupCli
{
    "AttachedPolicies": [
        {
            "PolicyName": "AmazonEC2FullAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"
        }
    ]
}
```

#### task6\_create\_login\_profile\_and\_signin.png

```
NayabKhzain653 eworkspaces/Lab-9 (main) $ aws iam create-login-profile --user-name MyUserCli
{
    "LoginProfile": {
        "UserName": "MyUserCli",
        "AccessKeyId": "AKIAUHXY42RXI2JEINEH",
        "Status": "Active",
        "SecretAccessKey": "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",
        "CreateDate": "2026-01-06T14:46:29+00:00"
    }
}
```

#### task6\_create\_access\_key\_output.png

```
NayabKhzain653 eworkspaces/Lab-9 (main) $ aws iam list-access-keys --user-name MyUserCli
{
    "AccessKeyMetadata": [
        {
            "UserName": "MyUserCli",
            "AccessKeyId": "AKIAUHXY42RXI2JEINEH",
            "Status": "Active",
            "CreateDate": "2026-01-06T14:46:29+00:00"
        }
    ]
}
```

#### task6\_create\_access\_key\_output.png

```
NayabKhzain653 eworkspaces/Lab-9 (main) $ export AWS_SECRET_ACCESS_KEY=A2/3W0VEV3hNAWyh/rYVD2PjwlBW/7PbueIQpm53
NayabKhzain653 eworkspaces/Lab-9 (main) $ printenv | grep AWS_
AWS_SECRET_ACCESS_KEY=A2/3W0VEV3hNAWyh/rYVD2PjwlBW/7PbueIQpm53
AWS_ACCESS_KEY_ID=AKIAUHXY42RXI2JEINEH
NayabKhzain653 eworkspaces/Lab-9 (main) $ aws iam get-user --user-name myUserCli
An error occurred (AccessDenied) when calling the GetUser operation: User: arn:aws:iam::291506017390:user/MyUserCli is not authorized to perform: iam:GetUser on resource
: user/MyUserCli because no identity-based policy allows the iam:GetUser action
```

#### task6\_env\_exports\_and\_get\_user\_error.png

## Task 7 — Filters: query with filters to find instances and their attributes

```
> --filters "Name>tag:Name,Values=MyServer" \
> --query "Reservations[*].Instances[*].PublicIpAddress" \
> --output text
40.172.230.52
```

#### task7\_filter\_by\_instance\_type.png

```
> --filters "Name=instance-type,Values=t3.micro" \
> --query "Reservations[].Instances[].InstanceId" \
> --output table
+-----+
| DescribeInstances |
+-----+
| i-0403609d04099cfc |
+-----+
```

task7\_filter\_by\_subnet.png

```
> --filters "Name=subnet-id,Values= subnet-05d6cfad232c55c6f" \
> --query "Reservations[*].Instances[*].InstanceId" \
> --output table
+-----+
| DescribeInstances |
+-----+
| i-0403609d04099cfc |
+-----+
```

task7\_filter\_by\_vpc.png

```
> --filters "Name=vpc-id,Values= vpc-0bfbc7302dc628797" \
> --query "Reservations[*].Instances[*].InstanceId" \
> --output table
+-----+
| DescribeInstances |
+-----+
| i-0403609d04099cfc |
+-----+
```

## Task 8 — Use --query to format outputs for reporting

task8\_query\_table\_instances\_name\_ip.png

```
> --filters "Name>tag:Name,Values=MyServer" \
> --query "Reservations[*].Instances*[].[InstanceId,PublicIpAddress,Tags[?Key=='Name'].Value|[0]]" \
> --output table
+-----+
| DescribeInstances |
+-----+
| i-0403609d04099cfc | 40.172.230.52 | MyServer |
+-----+
```

task8\_query\_table\_instance\_state.png

```
> --query "Reservations[*].Instances*[].[InstanceId,State.Name]" \
> --output table
+-----+
| DescribeInstances |
+-----+
| i-0403609d04099cfc | running |
+-----+
```

task8\_query\_table\_instance\_type\_az.png

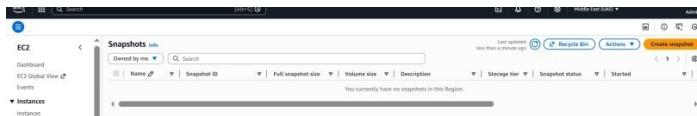
```
> --query "Reservations[*].Instances*[].[InstanceId,InstanceType,Placement.AvailabilityZone]" \
> --output table
+-----+
| DescribeInstances |
+-----+
| i-0403609d04099cfc | t3.micro | me-central-1b |
+-----+
```

## Cleanup — Remove resources to avoid charges

cleanup\_terminate\_instance.png

```
NayabKhazin653 [~/workspaces/Lab12 (main)] $ aws ec2 terminate-instances --instance-ids i-0403609d04099cbcb
{
  "TerminatingInstances": [
    {
      "InstanceId": "i-0403609d04099cbcb",
      "CurrentState": {
        "Code": 32,
        "Name": "shutting-down"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
```

cleanup\_delete\_volumes\_snapshots.png



cleanup\_delete\_security\_group\_and\_keypair.png

```
NayabKhazin653 [~/workspaces/Lab-9 (main)] $ aws ec2 delete-security-group --group-id sg-030a3daa25d097dca
{
  "Return": true,
  "GroupId": "sg-030a3daa25d097dca"
}
NayabKhazin653 [~/workspaces/Lab-9 (main)] $ aws ec2 delete-key-pair --key-name MyED25519Key
{
  "Return": true,
  "KeyPairId": "key-0d56d4d48d655a50f"
```

cleanup\_iam\_users\_deleted.png

```
NayabKhazin653 [~/workspaces/Lab12 (main)] $ aws iam delete-access-key --user-name MyUserCli --access-key-id AKIAZCUSI5S7GM4AUFLM
NayabKhazin653 [~/workspaces/Lab12 (main)] $ aws iam delete-login-profile --user-name MyUserCli
NayabKhazin653 [~/workspaces/Lab12 (main)] $ aws iam remove-user-from-group --user-name MyUserCli --group-name MyGroupCli
NayabKhazin653 [~/workspaces/Lab12 (main)] $ aws iam delete-user --user-name MyUserCli
NayabKhazin653 [~/workspaces/Lab12 (main)] $ aws iam detach-group-policy --group-name MyGroupCli --policy-arn arn:aws:iam::aws:policy/AmazonEC2FullAccess
NayabKhazin653 [~/workspaces/Lab12 (main)] $ aws iam delete-group --group-name MyGroupCli
```

cleanup\_summary.png

