

CLOUD COMPUTING LAB

Name: Nayab Khazin

Reg No: 2023-BSE-046

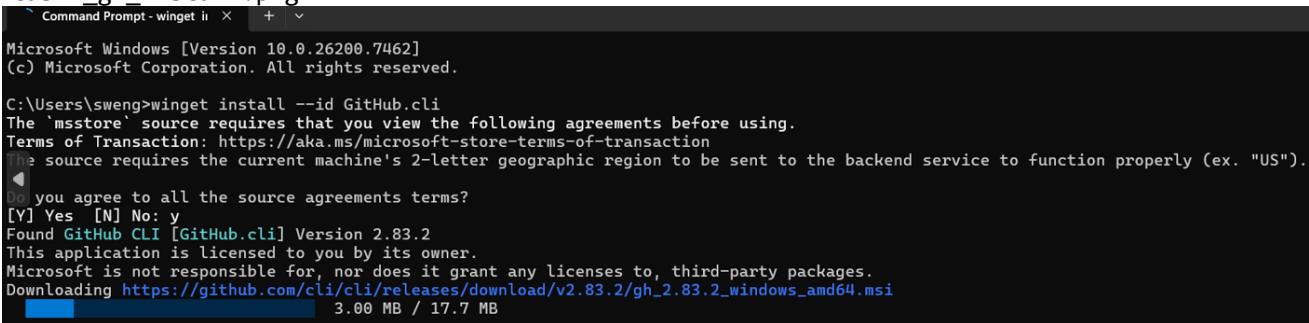
Submitted to: Sir Muhammad Shoaib

Class: 5B

LAB #09

Task 1 — GitHub CLI, Codespace setup and authentication

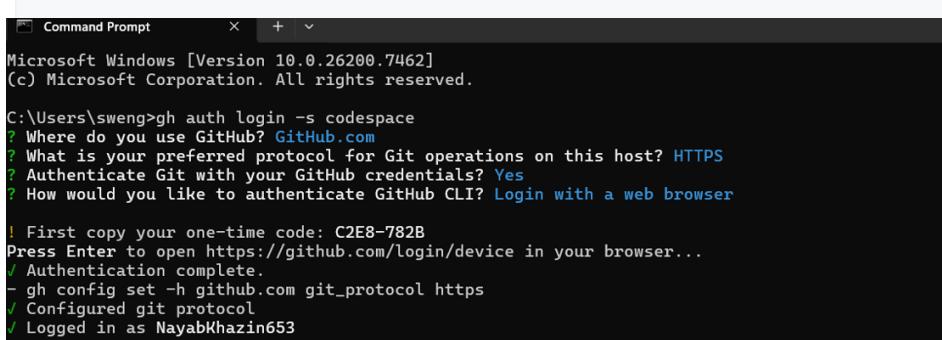
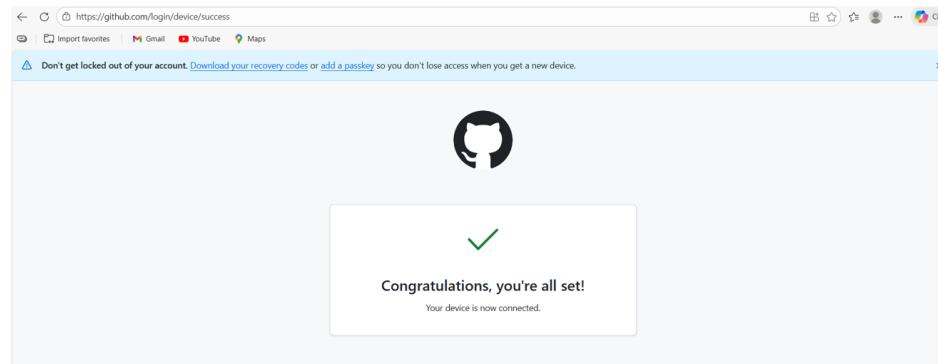
task1_gh_install.png



```
Command Prompt - winget i 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



```
Command Prompt
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.

@NayabKhazin653 → /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]: AKIAUHXY42RXNJIYIXKD6
AWS Secret Access Key [None]: Jx599I440pfT6bDE9EY8K0J7u4Gb5EIK2ZX6e3+X
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": "AIDAUHXY42RZF3IJLIPXL",
    "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-0e3a7ba238e097490",
            "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-0fac0d675ffaf878",
    "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",
                    "Status": "pending"
                }
            ],
            "State": "pending"
        }
    ]
}
```

task4_describe_instances_public_ip.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-instances --query "Reservations[*].Instances[*].PublicIpAddress,State.Name" --output table
-----|-----|-----|-----|
|     DescribeInstances      | |
|-----+-----+-----+-----|
| i-0fdbbb82cc7e99be09 | 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": [
                        ...
                    ]
                }
            ]
        }
    ]
}
```

task5_describe_vpcs.png

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws ec2 describe-vpcs
{
    "Vpcs": [
        {
            "OwnerId": "291506017390",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-0dec793abda69466",
                    "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-06a59b37715",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsRecord": 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-0e02d581cf0",
            "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.compute.amazonaws.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": [
                                "sg-0000000000000000"
                            ]
                        }
                    ]
                }
            ]
        }
    ]
}
```

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"
        }
    ]
}
```

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

task6_create_group_and_user.png

```
@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

```
@NayabKhazin653 → /workspaces/LAB-9 (main) $ aws iam create-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_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": "AIDAUHXY42RXNZ4ISGJZ",
            "Arn": "arn:aws:iam::291506017390:user/MyUserCli",
            "CreateDate": "2026-01-06T14:19:06+00:00"
        }
    ],
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAUHXY42RXNQMSVYMLT",
        "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
AWSLambdaBeanstalkCustomPlatformforEC2Role
AmazonEC2ContainerServiceEventsRole
AmazonEC2SpotFleetTaggingRole
AWSLambdaRoleforEC2ScheduledInstances
AWSFleetServiceRolePolicy
AWSApplicationAutoscalingEC2SpotFleetRequestPolicy
AWSFleetServiceRolePolicy
AWSAutoScalingPlansEC2AutoScalingPolicy
EC2InstanceConnect
AmazonEC2RolePolicyForLaunchWizard
EC2InstanceProfileForImageBuilder
EC2FleetTimeShiftableServiceRolePolicy
AmazonEC2RoleforAWSCodeDeployLimited
EC2InstanceProfileForImageBuilderECRContainerBuilds
AWSApplicationMigrationEC2Access
AWSComputeCapacityReservationFleetRolePolicy
EC2FastLaunchServiceRolePolicy
AmazonSSMManagedEC2InstanceDefaultPolicy
AWSFaultInjectionSimulatorEC2Access
EC2ImageBuilderLifecycleExecutionPolicy
AWSF2SSSnapshotPolicy
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

```
NayabKhazin653 @ /workspaces/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

```
NayabKhazin653 @ /workspaces/Lab-9 (main) $ aws iam attach-group-policy --group-name MyGroupCli --policy-arn arn:aws:iam::aws:policy/IAMUserChangePassword
NayabKhazin653 @ /workspaces/Lab-9 (main) $ aws iam detach-group-policy --group-name MyGroupCli --policy-arn arn:aws:iam::aws:policy/IAMUserChangePassword
```

task6_create_access_key_output.png

```
NayabKhazin653 @ /workspaces/Lab-9 (main) $ aws iam create-access-key --user-name MyUserCli
{
    "AccessKey": {
        "UserName": "MyUserCli",
        "AccessKeyId": "AKIAUHXY42RXI2JEINEH",
        "Status": "Active",
        "SecretAccessKey": "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",
        "CreateDate": "2026-01-06T14:46:29+00:00"
    }
}
```

task6_create_access_key_output.pn

```
NayabKhazin653 @ /workspaces/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

```
NayabKhazin653 @ /workspaces/Lab-9 (main) $ export AWS_ACCESS_KEY_ID=AKIAUHXY42RXI2JEINEH
NayabKhazin653 @ /workspaces/Lab-9 (main) $ export AWS_SECRET_ACCESS_KEY=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
NayabKhazin653 @ /workspaces/Lab-9 (main) $ printenv | grep AWS_
AWS_SECRET_ACCESS_KEY=A2/3W0VEV3hNAwYh/rYVD2PjwlBW/7PbueIQpm53
AWS_ACCESS_KEY_ID=AKIAUHXY42RXI2JEINEH
NayabKhazin653 @ /workspaces/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

task7_filter_by_tag_public_ip.png

```
> --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-0403609d04099cbb |
+-----+
```

task7_filter_by_subnet.png

```
> --filters "Name=subnet-id,Values= subnet-05d6cfad232c55c6f" \
> --query "Reservations[*].Instances[*].InstanceId" \
> --output table
| DescribeInstances |
+-----+
| i-0403609d04099cbb |
+-----+
```

task7_filter_by_vpc.png

```
> --filters "Name=vpc-id,Values= vpc-0bfbc7302dc628797" \
> --query "Reservations[*].Instances[*].InstanceId" \
> --output table
| DescribeInstances |
+-----+
| i-0403609d04099cbb |
+-----+
```

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-0403609d04099cbb | 40.172.230.52 | MyServer |
+-----+
```

task8_query_table_instance_state.png

```
> --query "Reservations[*].Instances[].[InstanceId,State.Name]" \
> --output table
| DescribeInstances |
+-----+
| i-0403609d04099cbb | running |
+-----+
```

task8_query_table_instance_type_az.png

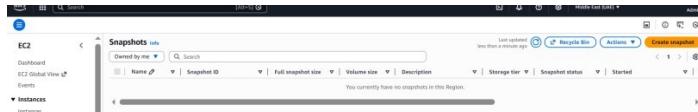
```
> --query "Reservations[*].Instances[].[InstanceId,InstanceType,Placement.AvailabilityZone]" \
> --output table
| DescribeInstances |
+-----+
| i-0403609d04099cbb | t3.micro | me-central-1b |
+-----+
```

Cleanup — Remove resources to avoid charges

cleanup_terminate_instance.png

```
@NayabKhazin653 ~/workspaces/Lab12 (main) $ aws ec2 terminate-instances --instance-ids i-0403609d04099cbc
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-0403609d04099cbc",
            "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

