Name: Sadaf Riaz

Roll No: 2023-BSE-077

Section : BSE5-B

Lab 09

Task 1 — GitHub CLI, Codespace setup and authentication

1.1 Install GH CLI (Windows example via winget) — Screenshot: task1_gh_install.png

```
PS C:\Users\Lenovo> winget install --id GitHub.cli
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.
Successfully verified installer hash
Starting package install...
Successfully installed
PS C:\Users\Lenovo>
```

1.2 Authenticate GH CLI for Codespaces — Screenshot: task1_gh_auth_login.png

```
PS C:\Users\Lenovo> 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 SadafRiaz-077
PS C:\Users\Lenovo>
```

1.3 List available Codespaces (optional verification) Screenshot:task1_codespace_list.png

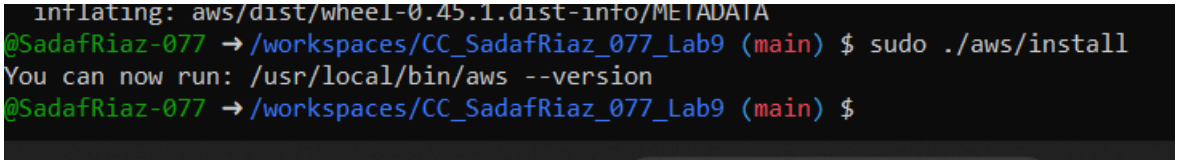
```
✓ Logged in as SadafRiaz-077
PS C:\Users\Lenovo> gh codespace list
NAME                          DISPLAY NAME          REPOSITORY                BRANCH  STATE
laughing-capybara-wrq7q...    laughing capybara     SadafRiaz-077/UbuntuMac...  main    Shutdown
Shutdown  about 28 days agoPS C:\Users\Lenovo>
```

1.4 Create or connect to a Codespace — Screenshot: task1_codespace_ssh_connected.png

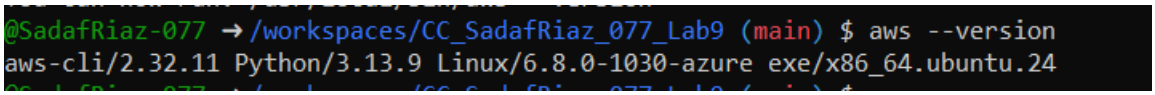
```
PS C:\Users\Lenovo\CC_SadafRiaz_077_Lab9> gh codespace create --repo "SadafRiaz-077/CC_SadafRiaz_077_Lab9" --branch main --ma
  ✓ Codespaces usage for this repository is paid for by SadafRiaz-077
obscure-fishstick-4jp7p65g5pjx2jqp9
PS C:\Users\Lenovo\CC_SadafRiaz_077_Lab9> gh codespace ssh -c "obscure-fishstick-4jp7p65g5pjx2jqp9"
Enter passphrase for key 'C:\Users\Lenovo/.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.

@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

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

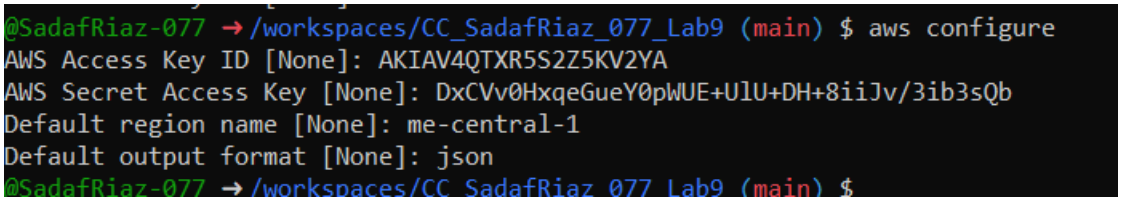
2.1 Download, unzip and install AWS CLI — Screenshot:task2_aws_install_and_version.png



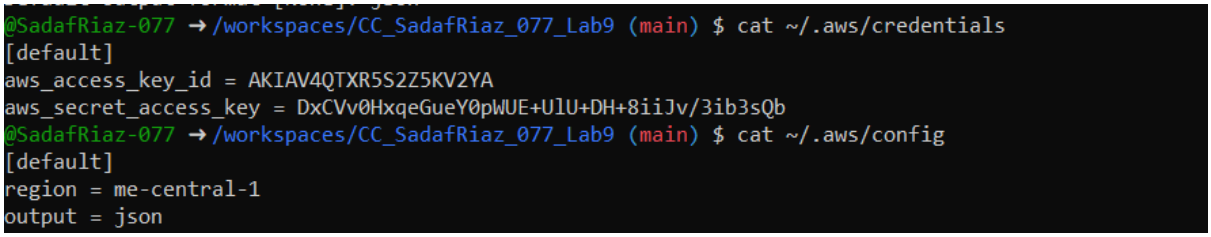
2.2 Verify installation (aws --version) — Screenshot: task2_aws_install_and_version.png



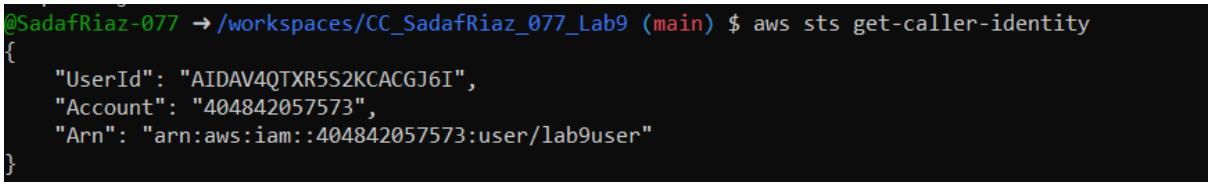
2.3 Configure AWS CLI (aws configure) — Screenshot: task2_aws_configure_and_files.png



2.4 Verify credentials/config files — Screenshot: task2_aws_configure_and_files.png

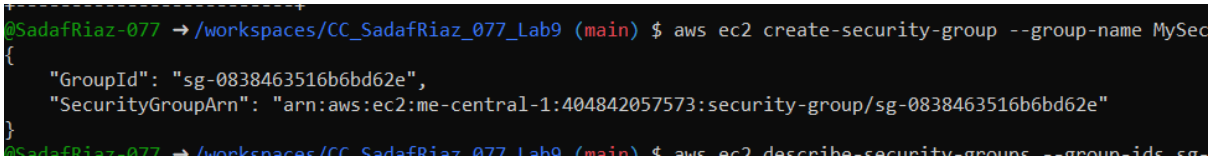


2.5 Verify connectivity (aws sts get-caller-identity) — Screenshot:task2_aws_get_caller_identity.png



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

3.1 Create security group — Screenshot: task3_create_security_group_output.png



3.2 Describe security group before adding rules — Screenshot: task3_describe_sg_before_ingress.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-security-groups --group-ids sg-0838463516b6
{
    "SecurityGroups": [
        {
            "GroupId": "sg-0838463516b6bd62e",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-0ed011693d93d59bc",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:404842057573:security-group/sg-0838463516b6bd62e",
            "OwnerId": "404842057573",
            "GroupName": "MySecurityGroup",
            "Description": "My Security Group",
            "IpPermissions": []
        }
    ]
}
```

3.3 Get Codespace public IP — Screenshot: task3_codespace_public_ip.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ curl icanhazip.com
20.192.21.54
```

3.4 Authorize SSH on port 22 — Screenshot: task3_authorize_ssh_and_describe.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 authorize-security-group-ingress \
> --group-id sg-0838463516b6bd62e \
> --protocol tcp \
> --port 22 \
> --cidr 20.192.21.54/32
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-03105bc230607c8e0",
            "GroupId": "sg-0838463516b6bd62e",
            "GroupOwnerId": "404842057573",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "20.192.21.54/32",
            "SecurityGroupRuleArn": "arn:aws:ec2:me-central-1:404842057573:security-group-rule/sgr-03105bc230607c
        }
    ]
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

3.5 Authorize HTTP on port 80 — Screenshot: task3_authorize_http_and_describe.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 authorize-security-group-ingres
> --group-id sg-0838463516b6bd62e \
> --ip-permissions '{"FromPort":80,"ToPort":80,"IpProtocol":"tcp","IpRanges":[{"CidrIp":"20.192.21.
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0198e648dbb03f3ef",
            "GroupId": "sg-0838463516b6bd62e",
            "GroupOwnerId": "404842057573",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "20.192.21.54/32",
            "SecurityGroupRuleArn": "arn:aws:ec2:me-central-1:404842057573:security-group-rule/sgr-
        }
    ]
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

3.6 Final describe showing both ingress rules — Screenshot: task3_describe_sg_final.png

```
        "GroupName": "MySecurityGroup",
        "Description": "My Security Group",
        "IpPermissions": [
            {
                "IpProtocol": "tcp",
                "FromPort": 80,
                "ToPort": 80,
                "UserIdGroupPairs": [],
                "IpRanges": [
                    {
                        "CidrIp": "20.192.21.54/32"
                    }
                ],
                "Ipv6Ranges": [],
                "PrefixListIds": []
            },
            {
                "IpProtocol": "tcp",
                "FromPort": 22,
                "ToPort": 22,
                "UserIdGroupPairs": [],
                "IpRanges": [
                    {
                        "CidrIp": "20.192.21.54/32"
                    }
                ],
                "Ipv6Ranges": [],
                "PrefixListIds": []
file is already in use  (press RETURN)
```

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

4.1 Create key pair (MyED25519Key.pem) — Screenshot: task4_create_keypair_output.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 create-key-pair \
>   --key-name MyED25519Key \
>   --key-type ed25519 \
>   --key-format pem \
>   --query 'KeyMaterial' \
>   --output text > MyED25519Key.pem
```

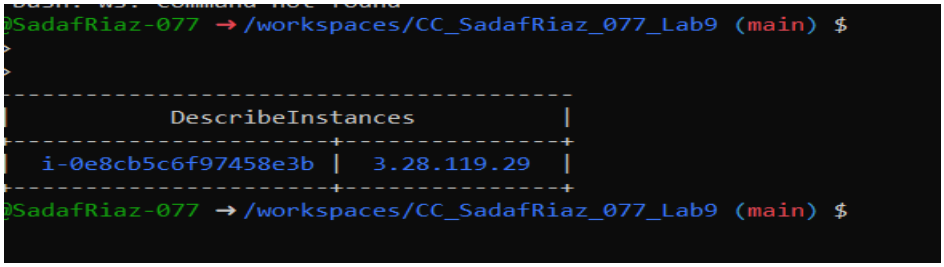
4.2 Describe key pairs — Screenshot: task4_describe_keypairs.png

```
-rw-rw-rw- 1 codespace codespace 0 Dec  8 09:31 MyED25519Key.pem
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-key-pairs
{
    "KeyPairs": [
        {
            "KeyPairId": "key-0ebbb2f91b611f468",
            "KeyType": "ed25519",
            "Tags": [],
            "CreateTime": "2025-12-06T18:45:38.269000+00:00",
            "KeyName": "Lab8Key",
            "KeyFingerprint": "bXc7beq4xqwmC7MP76sVQUYV4XQIM30cuvZu/VW7af0="
        },
        {
            "KeyPairId": "key-0942df99b41b66d5e",
            "KeyType": "ed25519",
            "Tags": [],
            "CreateTime": "2025-12-08T09:17:59.270000+00:00",
            "KeyName": "MyED25519Key",
            "KeyFingerprint": "4czFR5Ji3siOboEIt28MUa0bmpaNRQK7P9QrYMrcFps="
        }
    ]
}
```

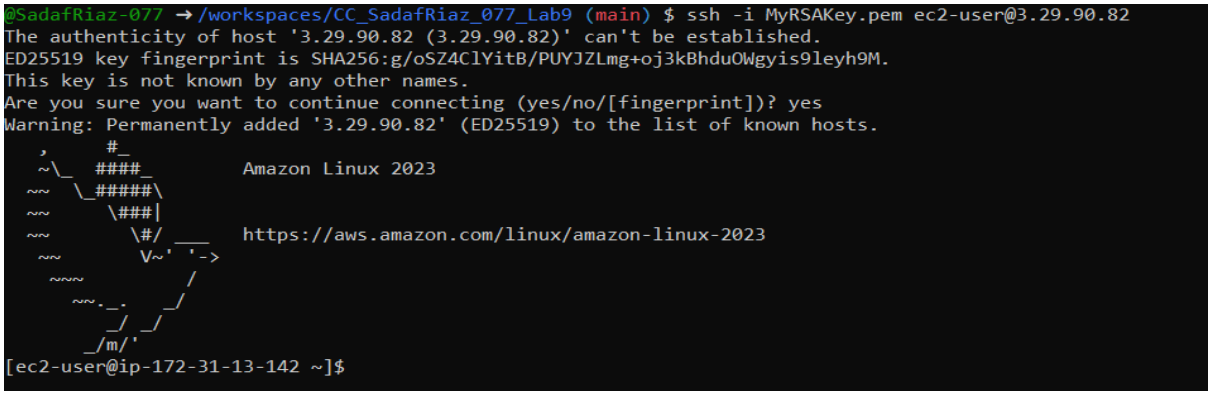
4.4 Launch EC2 instance — Screenshot: task4_run_instances_output.png


```
"Instances": [
    {
        "Architecture": "x86_64",
        "BlockDeviceMappings": [],
        "ClientToken": "a6a28eb4-4452-432c-8a66-c3edba4ff7e6",
        "EbsOptimized": false,
        "EnaSupport": true,
        "Hypervisor": "xen",
        "NetworkInterfaces": [
            {
                "Attachment": {
                    "AttachTime": "2025-12-08T09:45:37+00:00",
                    "AttachmentId": "eni-attach-08fd60a9f3b8c0a7b",
                    "DeleteOnTermination": true,
                    "DeviceIndex": 0,
                    "Status": "attaching",
                    "NetworkCardIndex": 0
                },
                "Description": "",
                "Groups": [
                    {
                        "GroupId": "sg-0838463516b6bd62e",
                        "GroupName": "MySecurityGroup"
                    }
                ],
```

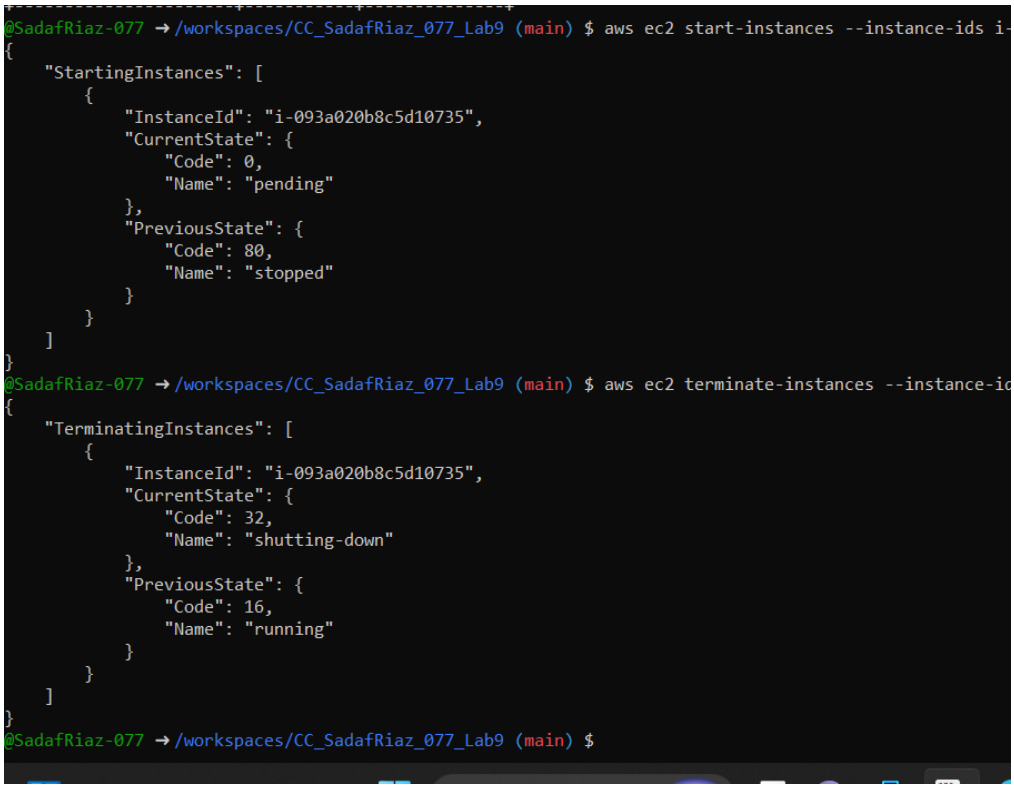
4.5 Describe instance public IP — Screenshot: task4_describe_instances_public_ip.png



4.6 SSH permission error & fix (chmod 400) — Screenshot: task4_ssh_permission_error_and_fix.png



4.7 Stop/start/terminate instance commands — Screenshot: task4_stop_start_terminate_commands.png



*Task 5 — Understand AWS describe- commands**

5.1 Describe security groups — Screenshot: task5_describe_security_groups.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-security-groups --no-cli-pager
{
    "SecurityGroups": [
        {
            "GroupId": "sg-07faa232dd28e816b",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-0ed011693d93d59bc",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:404842057573:security-group/sg-07faa232dd28e816b",
            "OwnerId": "404842057573",
            "GroupName": "default",
            "Description": "default VPC security group",
            "IpPermissions": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [
                        {
                            "UserId": "404842057573",
                            "GroupId": "sg-07faa232dd28e816b"
                        }
                    ],
                    "IpRanges": [],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ]
        },
        {
            "GroupId": "sg-09b089c5751d8822a",
            "IpPermissionsEgress": [
                {
```

5.2 Describe VPCs — Screenshot: task5_describe_vpcs.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-vpcs --no-cli-pager
{
    "Vpcs": [
        {
            "OwnerId": "404842057573",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-0fc13ce2206ae6fe2",
                    "CidrBlock": "172.31.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": true,
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "VpcId": "vpc-0ed011693d93d59bc",
            "State": "available",
            "CidrBlock": "172.31.0.0/16",
            "DhcpOptionsId": "dopt-00025b89642cbf410"
        }
    ]
}
```

5.3 Describe subnets — Screenshot: task5_describe_subnets.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-subnets --no-cli-pager
{
    "Subnets": [
        {
            "AvailabilityZoneId": "mec1-az3",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "404842057573",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "SubnetArn": "arn:aws:ec2:me-central-1:404842057573:subnet/subnet-04141434a57a42925",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            },
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "SubnetId": "subnet-04141434a57a42925",
            "State": "available",
            "VpcId": "vpc-0ed011693d93d59bc",
            "CidrBlock": "172.31.0.0/20",
            "AvailableIpAddressCount": 4090,
            "AvailabilityZone": "me-central-1c",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true
        },
        {
            "AvailabilityZoneId": "mec1-az2",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "404842057573",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "SubnetArn": "arn:aws:ec2:me-central-1:404842057573:subnet/subnet-023cec070fa4154f2",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
```

5.4 Describe instances — Screenshot: task5_describe_instances.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-instances --no-cli-pager
{
    "Reservations": [
        {
            "ReservationId": "r-0f3559f6f442e24f4",
            "OwnerId": "404842057573",
            "Groups": [],
            "Instances": [
                {
                    "Architecture": "x86_64",
                    "BlockDeviceMappings": [
                        {
                            "DeviceName": "/dev/xvda",
                            "Ebs": {
                                "AttachTime": "2025-12-08T09:45:38+00:00",
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-0db5afc9ae454e9cc"
                            }
                        }
                    ],
                    "ClientToken": "a6a28eb4-4452-432c-8a66-c3edba4ff7e6",
                    "EbsOptimized": false,
                    "EnaSupport": true,
                    "Hypervisor": "xen",
                    "NetworkInterfaces": [
                        {
                            "Association": {
                                "IpOwnerId": "amazon",
                                "PublicDnsName": "ec2-3-28-119-29.me-central-1.compute.amazonaws.com",
                                "PublicIp": "3.28.119.29"
                            },
```

5.5 Describe regions — Screenshot: task5_describe_regions.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-regions --no-cli-pager
{
    "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"
```

5.6 Describe availability zones — Screenshot: task5_describe_availability_zones.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-availability-zones --no-cl
{
    "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 6 — IAM: create group/user, attach policies, create login & keys

6.1 Create group & user — Screenshot: task6_create_group_and_user.png

```
                    ],
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam create-group --group-name MyGroupCli
{
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAV4QTXR5S3K2FR2TZ4",
        "Arn": "arn:aws:iam::404842057573:group/MyGroupCli",
        "CreateDate": "2025-12-08T12:52:32+00:00"
    }
}
```

6.2 Add user to group and verify — Screenshot: task6_add_user_to_group_and_verify.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam get-group --group-name MyGroupCli
{
    "Users": [],
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAV4QTXR5S3K2FR2TZ4",
        "Arn": "arn:aws:iam::404842057573:group/MyGroupCli",
        "CreateDate": "2025-12-08T12:52:32+00:00"
    }
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam create-user --user-name MyUserCli
{
    "User": {
        "Path": "/",
        "UserName": "MyUserCli",
        "UserId": "AIDAV4QTXR5SYJUG3FL5V",
        "Arn": "arn:aws:iam::404842057573:user/MyUserCli",
        "CreateDate": "2025-12-08T13:02:55+00:00"
    }
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam get-user --user-name MyUserCli
{
    "User": {
        "Path": "/",
        "UserName": "MyUserCli",
        "UserId": "AIDAV4QTXR5SYJUG3FL5V",
        "Arn": "arn:aws:iam::404842057573:user/MyUserCli",
        "CreateDate": "2025-12-08T13:02:55+00:00"
    }
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam add-user-to-group --user-name MyUserCli --group-name MyGroupCli
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

6.3 List & attach EC2-related policies — Screenshot: task6_policy_list_and_attach.png

```
AmazonEC2ContainerRegistryPullOnly
DeclarativePoliciesEC2Report
AmazonEC2ImageReferencesAccessPolicy
AWSEC2CapacityManagerServiceRolePolicy
AWSEC2SqlHaServiceRolePolicy
AWSEC2SqlHaInstancePolicy
AWSLambdaManagedEC2ResourceOperator
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam list-policies --query 'Policies[?PolicyName==`AmazonEC2FullAccess`].{Name:PolicyName, ARN:Arn}' --out
-------------------------------------------------------------------------
|                           ListPolicies                                |
+-----------------------------------------------+-----------------------+
|                      ARN                      |         Name          |
+-----------------------------------------------+-----------------------+
|  arn:aws:iam::aws:policy/AmazonEC2FullAccess  |  AmazonEC2FullAccess  |
+-----------------------------------------------+-----------------------+
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam attach-group-policy \
>   --group-name MyGroupCli \
>   --policy-arn arn:aws:iam::aws:policy/AmazonEC2FullAccess
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

6.4 Create login profile & sign in — Screenshot: task6_create_login_profile_and_signin.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam list-attached-group-policies --group-name
{
    "AttachedPolicies": [
        {
            "PolicyName": "AmazonEC2FullAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"
        }
    ]
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam create-login-profile \
>   --user-name MyUserCli \
>   --password <PASSWORD_VALUE> \
>   --password-reset-required
-bash: PASSWORD_VALUE: No such file or directory
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

6.5 Create & list access keys — Screenshot: task6_create_access_key_output.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam attach-group-policy --group-name MyGroupCli --policy-arn arn:aws:iam::aws:policy/IAMUserC
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam detach-group-policy --group-name MyGroupCli --policy-arn arn:aws:iam::aws:policy/IAMUserC
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam create-access-key --user-name MyUserCli
{
    "AccessKey": {
        "UserName": "MyUserCli",
        "AccessKeyId": "AKIAV4QTXR5SSP55CZ6J",
        "Status": "Active",
        "SecretAccessKey": "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",
        "CreateDate": "2025-12-08T13:23:13+00:00"
    }
}
```

6.6 Test environment variable auth Screenshot:task6_env_exports_and_get_user_error.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ export AWS_ACCESS_KEY_ID=AKIAV4QTXR5SSP55CZ6J
port AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ printenv | grep AWS_
AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
AWS_ACCESS_KEY_ID=AKIAV4QTXR5SSP55CZ6J
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

6.7 After clearing creds, verify user — Screenshot: task6_after_logout_and_get_user_success.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam get-user --user-name MyUserCli

An error occurred (InvalidClientTokenId) when calling the GetUser operation: The security token included in the request is in
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ unset AWS_ACCESS_KEY_ID
WS_SECRET_ACCESS_KEY
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ unset AWS_SECRET_ACCESS_KEY
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam get-user --user-name MyUserCli
{
    "User": {
        "Path": "/",
        "UserName": "MyUserCli",
        "UserId": "AIDAV4QTXR5SYJUG3FL5V",
        "Arn": "arn:aws:iam::404842057573:user/MyUserCli",
        "CreateDate": "2025-12-08T13:02:55+00:00"
    }
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

Task 7 — Filters: query instances and attributes

7.1 Filter by tag (Public IP) — Screenshot: task7_filter_by_tag_public_ip.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-instances \
>    --filters "Name=tag:Name,Values=MyServer" \
>    --query "Reservations[*].Instances[*].PublicIpAddress" \
>    --output text
3.28.119.29
40.172.221.98
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

7.2 Filter by instance type — Screenshot: task7_filter_by_instance_type.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-instances \
>    --filters "Name=instance-type,Values=t3.micro" \
>    --query "Reservations[].Instances[].InstanceId" \
>    --output table
------------------------
|   DescribeInstances   |
+-----------------------+
|  i-0e8cb5c6f97458e3b  |
|  i-0c8be122aec9e2d17  |
+-----------------------+
```

7.3 Filter by subnet — Screenshot: task7_filter_by_subnet.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-instances \
>    --filters "Name=subnet-id,Values=subnet-04141434a57a42925" \
>    --query "Reservations[*].Instances[*].InstanceId" \
>    --output table
------------------------
|   DescribeInstances   |
+-----------------------+
|  i-0e8cb5c6f97458e3b  |
+-----------------------+
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

7.4 Filter by VPC — Screenshot: task7_filter_by_vpc.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-instances \
>    --filters "Name=vpc-id,Values=vpc-0ed011693d93d59bc" \
>    --query "Reservations[*].Instances[*].InstanceId" \
>    --output table
------------------------
|   DescribeInstances   |
+-----------------------+
|  i-0e8cb5c6f97458e3b  |
|  i-0c8be122aec9e2d17  |
+-----------------------+
```

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

8.1 Query table: InstanceId, PublicIp, Name — Screenshot: task8_query_table_instances_name_ip.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_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-0e8cb5c6f97458e3b |  3.28.119.29    |  MyServer |
|  i-0c8be122aec9e2d17 |  40.172.221.98  |  MyServer |
+----------------------+-----------------+-----------+
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

8.2 Query table: InstanceId & State — Screenshot: task8_query_table_instance_state.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-instances \
>   --query "Reservations[*].Instances[*].[InstanceId,State.Name]" \
>   --output table
-----------------------------------
|         DescribeInstances       |
+----------------------+----------+
|  i-0e8cb5c6f97458e3b |  running |
|  i-0c8be122aec9e2d17 |  running |
+----------------------+----------+
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

8.3 Query table: InstanceId, InstanceType, AvailabilityZone — Screenshot: task8_query_table_instance_type_az.png

```
+----------------------+----------+
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-instances \
>   --query "Reservations[*].Instances[*].[InstanceId,InstanceType,Placement.AvailabilityZone]" \
>   --output table
-----------------------------------------------------
|                 DescribeInstances                 |
+----------------------+-----------+----------------+
|  i-0e8cb5c6f97458e3b |  t3.micro |  me-central-1c |
|  i-0c8be122aec9e2d17 |  t3.micro |  me-central-1a |
+----------------------+-----------+----------------+
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

Cleanup — Remove resources to avoid charges

Cleanup 1 — Terminate instances — Screenshot: cleanup_terminate_instance.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 termir
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-0e8cb5c6f97458e3b",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        },
        {
            "InstanceId": "i-0c8be122aec9e2d17",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

Cleanup 2 — Delete EBS volumes & snapshots Screenshot:cleanup_delete_volumes_snapshots.png

```
SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-volumes --query "Volumes[*].[VolumeId,State]" --output table
SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 delete-volume --volume-id vol-XXXXXXXXXXXXXXX

n error occurred (InvalidParameterValue) when calling the DeleteVolume operation: The volume ID 'vol-XXXXXXXXXXXXXXX' is malformed
SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

Cleanup 3 — Delete security group & key pair — Screenshot: cleanup_delete_security_group_and_keypair.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 delete-security-group --group-id sg-09b089c5751d8822a
{
    "Return": true,
    "GroupId": "sg-09b089c5751d8822a"
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 delete-security-group --group-id sg-0838463516b6bd62e
{
    "Return": true,
    "GroupId": "sg-0838463516b6bd62e"
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-volumes --query "Volumes[*].[VolumeId,State]" --output table
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ ^C
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 delete-key-pair --key-name MyED25519Key
{
    "Return": true,
    "KeyPairId": "key-0942df99b41b66d5e"
}
```

Cleanup 4 — Remove IAM users, access keys, groups — Screenshot: cleanup_iam_users_deleted.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam list-access-keys --user-name MyUserCli
{
    "AccessKeyMetadata": []
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam delete-access-key --user-name MyUserCli --access-key-id AKIAV4QTXR5SSP55CZ6J

An error occurred (NoSuchEntity) when calling the DeleteAccessKey operation: The Access Key with id AKIAV4QTXR5SSP55CZ6J cannot be found.
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam remove-user-from-group --user-name MyUserCli --group-name MyGroupCli
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam remove-user-from-group --user-name MyUserCli --group-name MyGroupCli
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam delete-user --user-name MyUserCli
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam detach-group-policy --group-name MyGroupCli --policy-arn arn:aws:iam::aws:poli
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws iam delete-group --group-name MyGroupCli
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```

Cleanup 5 — Final verification (billing/resources) — Screenshot: cleanup_summary.png

```
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-instances --query "Reservations[*].Instances[*].[InstanceId,Sta
----------------------------------------
|          DescribeInstances           |
+----------------------+-------------+
|  i-0e8cb5c6f97458e3b |  terminated  |
|  i-0c8be122aec9e2d17 |  terminated  |
+----------------------+-------------+
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-security-groups --query "SecurityGroups[*].[GroupId,GroupName]"
-----------------------------------
|      DescribeSecurityGroups       |
+----------------------+-----------+
|  sg-07faa232dd28e816b |  default  |
+----------------------+-----------+
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-key-pairs --query "KeyPairs[*].[KeyName]" --output table
-----------------
|DescribeKeyPairs|
+----------------+
|  Lab8Key       |
|  MyRSAKey      |
+----------------+
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 delete-key-pair --key-name Lab8Key
 ec2 delete-key-pair --key-name MyRSAKey
{
    "Return": true,
    "KeyPairId": "key-0ebbb2f91b611f468"
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 delete-key-pair --key-name MyRSAKey
{
    "Return": true,
    "KeyPairId": "key-0fa3bc44f8ee0a97e"
}
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $ aws ec2 describe-key-pairs --query "KeyPairs[*].[KeyName]" --output table
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
@SadafRiaz-077 ➜ /workspaces/CC_SadafRiaz_077_Lab9 (main) $
```