

## **CLOUD COMPUTING**



**SUBMITTED TO**

ENGR. SHOAIB & SIR WAQAS SALEEM

**SUBMITTED BY**  
HAMNA MAHMOOD

2023-BSE-025

BSE V-A

## LAB 09

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

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

- (Local desktop) Install GitHub CLI:

task1\_gh\_install.png



```
C:\Users\ABC\Downloads\gh_2.83.2_windows_amd64\bin>gh --version
gh version 2.83.2 (2025-12-10)
https://github.com/cli/cli/releases/tag/v2.83.2
```

task1\_gh\_auth\_login.png

- (Local) Authenticate GH CLI for Codespaces:  
task1\_gh\_auth\_login.png

```
C:\Users\ABC\Downloads\gh_2.83.2_windows_amd64\bin>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 hamna-mahmood
```

- (Local) List available Codespaces (optional verification):  
task1\_codespace\_list.png

```
C:\Users\ABC\Downloads\gh_2.83.2_windows_amd64\bin>gh codespace list
no codespaces found
```

- (Local) Create or connect to a Codespace.  
task1\_codespace\_ssh\_connected.png

```
@hamna-mahmood [?] /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

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

- Download and install AWS CLI:

task2\_aws\_install\_and\_version.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  % Total    % Received % Xferd  Average Speed   Time     Time      Current
                                         Dload  Upload Total Spent   Left  Speed
100  60.2M  100  60.2M    0     0  170M      0 --:--:-- --:--:-- 170M
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ unzip awscliv2.zip
Archive: awscliv2.zip
  creating: aws/
  creating: aws/dist/
  extracting: aws/_dist/awscli wheel 0x00000000000000000000000000000000
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ sudo ./aws/install
You can now run: /usr/local/bin/aws --version
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws --version
aws-cli/2.32.26 Python/3.13.11 Linux/6.8.0-1030-azure exe/x86_64.ubuntu.24
```

- Verify installation:

task2\_aws\_install\_and\_version.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  % Total    % Received % Xferd  Average Speed   Time     Time      Current
                                         Dload  Upload Total Spent   Left  Speed
100  60.2M  100  60.2M    0     0  170M      0 --:--:-- --:--:-- 170M
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ unzip awscliv2.zip
Archive: awscliv2.zip
  creating: aws/
  creating: aws/dist/
  extracting: aws/_dist/awscli wheel 0x00000000000000000000000000000000
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ sudo ./aws/install
You can now run: /usr/local/bin/aws --version
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws --version
aws-cli/2.32.26 Python/3.13.11 Linux/6.8.0-1030-azure exe/x86_64.ubuntu.24
```

- Configure the AWS CLI (you will provide Access Key ID and Secret Access Key for a user with permissions, or use root/IAM user you prepared for the lab):

task2\_aws\_configure\_and\_files.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]
Default region name [None]: me-central-1
Default output format [None]: json
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

- Verify credentials/config files:

task2\_aws\_configure\_and\_files.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ cat ~/.aws/credentials
[default]
aws_access_key_id =
aws_secret_access_key
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ cat ~/.aws/config
[default]
region = me-central-1
output = json
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

- Verify connectivity (you should see a JSON result showing your caller identity):

task2\_aws\_get\_caller\_identity.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws sts get-caller-identity
{
    "UserId": "AIDAX4VWZI3PVPPIQABGB",
    "Account": "542622959327",
    "Arn": "arn:aws:iam::542622959327:user/Hamna_Assignment"
}
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

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

- Create a security group (substitute your VPC id):

task3\_create\_security\_group\_output.png

```
@hamna-mahmood 🐶 /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 create-security-group --group-name 'MySecurityGroup' \
>   --description 'My Security Group' \
>   --vpc-id 'vpc-09822723aed1d2815'
{
    "GroupId": "sg-0e075b42f5a5ce11e",
    "SecurityGroupArn": "arn:aws:ec2:me-central-1:542622959327:security-group/sg-0e075b42f5a5ce11e"
}
```

- Inspect the security group (initially ingress will be empty or default):

task3\_describe\_sg\_before\_ingress.png

```
@hamna-mahmood 🐶 /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-security-groups --group-ids sg-0e075b42f5a5ce11e
{
    "SecurityGroups": [
        {
            "GroupId": "sg-0e075b42f5a5ce11e",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-09822723aed1d2815",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:542622959327:security-group/sg-0e075b42f5a5ce11e",
            "OwnerId": "542622959327",
            "GroupName": "MySecurityGroup",
            "Description": "My Security Group",
            "IpPermissions": []
        }
    ]
}
```

- Get your Codespace public IP (from inside the Codespace):

task3\_codespace\_public\_ip.png

```
@hamna-mahmood 🐶 /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ curl icanhazip.com
4.240.18.225
```

- Authorize SSH inbound on port 22 from your Codespace IP:

task3\_authorize\_ssh\_and\_describe.png

```
@hamna-mahmood 🐶 /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 authorize-security-group-ingress --group-id sg-0e075b42f5a5ce11e \
--protocol tcp --port 22 --cidr 4.240.18.225/32
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-05a7872b75e3625f6",
            "GroupId": "sg-0e075b42f5a5ce11e",
            "GroupOwnerId": "542622959327",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "4.240.18.225/32",
            "SecurityGroupRuleArn": "arn:aws:ec2:me-central-1:542622959327:security-group-rule/sgr-05a7872b75e3625f6"
    ]
}
```

- Verify ingress rule appears:

task3\_authorize\_ssh\_and\_describe.png

```
@hamna-mahmood 🐶 /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-security-groups --group-ids sg-0e075b42f5a5ce11e
{
    "SecurityGroups": [
        {
            "GroupId": "sg-0e075b42f5a5ce11e",
            "IpPermissionsEgress": [
                {

```

- Add an HTTP rule (port 80) using ip-permissions JSON:

task3\_authorize\_http\_and\_describe.png

```
@hamna-mahmood ~ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 authorize-security-group-ingress --group-id 'sg-0e075b42f5a5ce11e' --ip-permissions '{"FromPort":80,"ToPort":80,"IpProtocol":"tcp","IpRanges":[{"CidrIp":"2.240.18.225/32"}]}'
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0149b08e7377ef879",
            "GroupId": "sg-0e075b42f5a5ce11e",
            "GroupOwnerId": "542622959327",
            "isEgress": false,
            "ipProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "2.240.18.225/32",
            "securityGroupRuleArn": "arn:aws:ec2:me-central-1:542622959327:security-group-rule/sgr-0149b08e7377ef879"
        }
    ]
}
@hamna-mahmood ~ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

- Verify both ingress rules are present:

task3\_describe\_sg\_final.png

```
@hamna-mahmood ~ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-security-groups --group-ids sg-0e075b42f5a5ce11e
{
    "SecurityGroups": [
        {
            "GroupId": "sg-0e075b42f5a5ce11e",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "VpcId": "vpc-09822723aed1d2815",
            "SecurityGroupArn": "arn:aws:ec2:me-central-1:542622959327:security-group/sg-0e075b42f5a5ce11e",
            "OwnerId": "542622959327",
            "GroupName": "MySecurityGroup",
            "Description": "My Security Group",
            "IpPermissions": [
                {
                    "IpProtocol": "tcp",
                    "FromPort": 80,
                    "ToPort": 80,
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "2.240.18.225/32"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                },
                {
                    "IpProtocol": "tcp",
                    "FromPort": 22,
                    "ToPort": 22,
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ]
        }
    ]
}
```

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

- Create the key pair and save the PEM file into the Codespace workspace:

task4\_create\_keypair\_output.png

```
@hamna-mahmood ~ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 create-key-pair \
>   --key-name MyED25519Key \
>   --key-type ed25519 \
>   --key-format pem \
>   --query 'KeyMaterial' \
>   --output text > MyED25519Key.pem
@hamna-mahmood ~ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ ls -l MyED25519Key.pem
-rw-rw-rw- 1 codespace codespace 388 Jan  2 20:48 MyED25519Key.pem
@hamna-mahmood ~ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

- View created key pairs:

task4\_describe\_keypairs.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-key-pairs
{
    "KeyPairs": [
        {
            "KeyId": "key-082c144ac8c3d074c",
            "KeyType": "ed25519",
            "Tags": [],
            "CreateTime": "2025-12-30T17:09:02.769000+00:00",
            "KeyName": "web-3-3-key",
            "KeyFingerprint": "VmsrVcKxmSslZ73bwBnB4jhI8xLHGDy044VZQaL7gt4="
        },
        {
            "KeyId": "key-01a87edae3f0198fe",
            ...
        }
    ]
}
```

- Launch an EC2 instance (example values — replace IDs with ones from your account/region):

task4\_run\_instances\_output.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 run-instances --image-id ami-05e66df2bafcb7dea --count 1 --instance-type t3.micro --key-name MyED25519Key --security-group-ids sg-0e075b42f5a5ce11e --subnet-id subnet-0a2cee75f0ad691a6 --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=MyServer}]"
{
    "ReservationId": "r-0d538521c04eb99a4",
    "OwnerId": "542622959327",
    "Groups": [],
    "Instances": [
        {
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "4bfd16d6-8cb3-4459-ab69-c671c1f5c60e",
            "EbsOptimized": false,
            "EnaSupport": true,
            ...
        }
    ]
}
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 run-instances --image-id ami-05e66df2bafcb7dea --count 1 --instance-type t3.micro --key-name MyED25519Key --security-group-ids sg-0e075b42f5a5ce11e --subnet-id subnet-0a2cee75f0ad691a6 --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=MyServer}]" --query "Instances[0].InstanceId" --output text
i-00a608cde40fd0cf1
```

- Get the public IP address of your instance:

task4\_describe\_instances\_public\_ip.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-instances \
> --query "Reservations[*].Instances[*].[InstanceId,PublicIpAddress]" \
> --output table
-----
|      DescribeInstances      |
+-----+
| i-0fa9d64afed5822d6 | 3.28.133.115 |
| i-01b0026b0394a0b56 | 3.29.33.6   |
| i-09dc2f89d7b6a07a5 | 158.252.33.74 |
| i-06a6d1c601e3e9e60 | 3.29.125.165 |
| i-01d026d26fc0ea152 | 3.28.135.154 |
| i-00a608cde40fd0cf1 | 3.28.207.172 |
+-----+
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

- Attempt SSH into the instance from the Codespace or from a machine whose IP is allowed in the security group:

task4\_ssh\_permission\_error\_and\_fix.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ ssh -i MyED25519Key.pem ec2-user@40.172.101.30
The authenticity of host '40.172.101.30 (40.172.101.30)' can't be established.
ED25519 key fingerprint is SHA256:CHKy9kfrR2EbUU3kemPfrDTgXe+BKfLVC0eNHgMadbs.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '40.172.101.30' (ED25519) to the list of known hosts.

          #
 ~\_ #####      Amazon Linux 2023
 ~~ \#####\
 ~~  \###|
 ~~   \#/   https://aws.amazon.com/linux/amazon-linux-2023
 ~~   V~,'-->
 ~~   /
 ~~.~/_/~/_
 ~~ /m/` 

[ec2-user@ip-10-0-10-102 ~]$
```

- Stop, start and (optionally) terminate the instance:

task4\_stop\_start\_terminate\_commands.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws ec2 stop-instances --instance-ids i-040929aa82fc20cce
{
  "StoppingInstances": [
    {
      "InstanceId": "i-040929aa82fc20cce",
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
[ec2-user@ip-10-0-10-102 ~]$ Connection to 40.172.101.30 closed by remote host.
Connection to 40.172.101.30 closed.
Hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 start-instances --instance-ids i-040929aa82fc20cce
{
  "StartingInstances": [
    {
      "InstanceId": "i-040929aa82fc20cce",
      "CurrentState": {
        "Code": 8,
        "Name": "pending"
      },
      "PreviousState": {
        "Code": 88,
        "Name": "stopped"
      }
    }
  ]
}
Hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-instances \
--instance-ids i-040929aa82fc20cce \
--query "Reservations[0].Instances[0].State.Name" \
--output text
running
Hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

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

Run and understand these commands (run each, then capture screenshot immediately after):  
 task5\_describe\_security\_groups.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws ec2 describe-security-groups
{
    "SecurityGroups": [
        {
            "GroupId": "sg-06d1566c80bccf8ad",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "tcp",
                    "FromPort": 80,
                    "ToPort": 80,
                    "UserIdGroupPairs": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": []
                }
            ],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "prod-backend-sg"
                }
            ],
            "VpcId": "vpc-09822723aed1d2815",
            "...skipping...

```

task5\_describe\_vpcs.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws ec2 describe-vpcs
{
    "Vpcs": [
        {
            "OwnerId": "542622959327",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-05694f8e33f894621",
                    "CidrBlock": "10.0.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": false,
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "prod-vpc"
                }
            ],
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "VpcId": "vpc-09822723aed1d2815",
            "State": "available",
            "CidrBlock": "10.0.0.0/16",
            "DhcpOptionsId": "dopt-073b0c55c0a106d5a"
        },
        {
            "OwnerId": "542622959327",
            "InstanceTenancy": "default",

```

task5\_describe\_subnets.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws ec2 describe-subnets
{
    "Subnets": [
        {
            "AvailabilityZoneId": "mec1-az2",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "542622959327",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "SubnetArn": "arn:aws:ec2:me-central-1:542622959327:subnet/subnet-078f1b79825a5fee0",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            },
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "SubnetId": "subnet-078f1b79825a5fee0",
            "State": "available",
            "VpcId": "vpc-0b412746b28b797e7",
            "CidrBlock": "172.31.16.0/20",
            "AvailableIpAddressCount": 4091,
            "AvailabilityZone": "me-central-1b",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true
        },
        {
            "AvailabilityZoneId": "mec1-az1",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "542622959327",
            "SubnetId": "subnet-078f1b79825a5fee1",
            "State": "available",
            "VpcId": "vpc-0b412746b28b797e7",
            "CidrBlock": "172.31.17.0/20",
            "AvailableIpAddressCount": 4091,
            "AvailabilityZone": "me-central-1a",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true
        }
    ]
}
```

task5\_describe\_instances.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws ec2 describe-instances
{
    "Reservations": [
        {
            "ReservationId": "r-070230ab471c7c75d",
            "OwnerId": "542622959327",
            "Groups": [],
            "Instances": [
                {
                    "Architecture": "x86_64",
                    "BlockDeviceMappings": [
                        {
                            "DeviceName": "/dev/xvda",
                            "Ebs": {
                                "AttachTime": "2025-12-30T17:18:51+00:00",
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-085ca9941c4724e91"
                            }
                        }
                    ],
                    "ClientToken": "terraform-20251230171846486800000002",
                    "EbsOptimized": false,
                    "EnaSupport": true,
                    "Hypervisor": "xen",
                    "NetworkInterfaces": [
                        {
                            "Association": {
                                "IpOwnerId": "amazon",
                                "PublicDnsName": "",
                                "PublicIp": "3.28.133.115"
                            },
                            "Attachment": {
                                "AttachTime": "2025-12-30T17:18:50+00:00"
                            }
                        }
                    ],
                    "RootDeviceType": "ebs",
                    "State": "running",
                    "Type": "t2.micro"
                }
            ]
        }
    ]
}
```

task5\_describe\_regions.png

```
[ec2-user@ip-10-0-10-102 ~]$ 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

```
[ec2-user@ip-10-0-10-102 ~]$ 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",
            "....skipping...
        },
        {
            "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"
                }
            ]
        }
    ]
}
```

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

Create group:

task6\_create\_group\_and\_user.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam create-group --group-name MyGroupCli
{
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAX4VWZI3PRLQX2CJWF",
        "Arn": "arn:aws:iam::542622959327:group/MyGroupCli",
        "CreateDate": "2026-01-04T12:28:12+00:00"
    }
}
[ec2-user@ip-10-0-10-102 ~]$
```

Get group details:

task6\_create\_group\_and\_user.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam get-group --group-name MyGroupCli
{
    "Users": [],
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAX4VWZI3PRLQX2CJWF",
        "Arn": "arn:aws:iam::542622959327:group/MyGroupCli",
        "CreateDate": "2026-01-04T12:28:12+00:00"
    }
}
[ec2-user@ip-10-0-10-102 ~]$
```

Create user:

task6\_create\_group\_and\_user.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam create-user --user-name MyUserCli
{
    "User": {
        "Path": "/",
        "UserName": "MyUserCli",
        "UserId": "AIDAX4VWZI3PYBIHCD3HA",
        "Arn": "arn:aws:iam::542622959327:user/MyUserCli",
        "CreateDate": "2026-01-04T12:30:34+00:00"
    }
}
[ec2-user@ip-10-0-10-102 ~]$
```

Get user details:

task6\_create\_group\_and\_user.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam add-user-to-group --user-name MyUserCli --group-name MyGroupCli
[ec2-user@ip-10-0-10-102 ~]$
```

Add user to group:

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

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam add-user-to-group --user-name MyUserCli --group-name MyGroupCli
[ec2-user@ip-10-0-10-102 ~]$
```

See group again:

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

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam get-group --group-name MyGroupCli
{
    "Users": [
        {
            "Path": "/",
            "UserName": "MyUserCli",
            "UserId": "AIDAX4VWZI3PYBIHCD3HA",
            "Arn": "arn:aws:iam::542622959327:user/MyUserCli",
            "CreateDate": "2026-01-04T12:30:34+00:00"
        }
    ],
    "Group": {
        "Path": "/",
        "GroupName": "MyGroupCli",
        "GroupId": "AGPAX4VWZI3PRLQX2CJWF",
        "Arn": "arn:aws:iam::542622959327:group/MyGroupCli",
        "CreateDate": "2026-01-04T12:28:12+00:00"
    }
}
[ec2-user@ip-10-0-10-102 ~]$
```

List policies that mention EC2:

task6\_policy\_list\_and\_attach.png

```
[ec2-user@ip-10-0-10-102 ~]$ 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
AWSEC2SpotServiceRolePolicy
AWSserviceRoleForEC2ScheduledInstances
:....skipping...
AmazonEC2FullAccess
AmazonEC2ReadOnlyAccess
AmazonElasticMapReduceforEC2Role
AmazonEC2RoleforDataPipelineRole
AmazonEC2ContainerServiceforEC2Role
AmazonEC2ContainerServiceRole
AmazonEC2RoleforAWSCodeDeploy
AmazonEC2RoleforSSM
CloudWatchActionsEC2Access
AmazonEC2ContainerRegistryReadOnly
AmazonEC2ContainerRegistryPowerUser
AmazonEC2ContainerRegistryFullAccess
AmazonEC2ContainerServiceAutoscaleRole
AmazonEC2SpotFleetAutoscaleRole
AWSelasticBeanstalkCustomPlatformforEC2Role
AmazonEC2ContainerServiceEventsRole
AmazonEC2SpotFleetTaggingRole
AWSEC2SpotServiceRolePolicy
AWSserviceRoleForEC2ScheduledInstances
AWSEC2SpotFleetServiceRolePolicy
```

Get ARN for AmazonEC2FullAccess (example query):

task6\_policy\_list\_and\_attach.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam list-policies --query 'Policies[?PolicyName==`AmazonEC2FullAccess`].{Name:PolicyName, ARN:Arn}' --output table
|-----+-----+
|      ListPolicies      |
|-----+-----+
|      ARN            |      Name        |
|-----+-----+
| arn:aws:iam::aws:policy/AmazonEC2FullAccess | AmazonEC2FullAccess |
|-----+-----+
[ec2-user@ip-10-0-10-102 ~]$
```

Attach policy to group (use the ARN you retrieved):

task6\_policy\_list\_and\_attach.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam attach-group-policy --group-name MyGroupCli --policy-arm arn:aws:iam::aws:policy/AmazonEC2FullAccess
[ec2-user@ip-10-0-10-102 ~]$
```

List attached policies for the group:

task6\_policy\_list\_and\_attach.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam list-attached-group-policies --group-name MyGroupCli
{
    "AttachedPolicies": [
        {
            "PolicyName": "AmazonEC2FullAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"
        }
    ]
}
[ec2-user@ip-10-0-10-102 ~]$
```

Create a console login profile for the user:

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

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam create-login-profile --user-name MyUserCLI --password [REDACTED] --password-reset-required
{
    "LoginProfile": {
        "UserName": "MyUserCLI",
        "CreateDate": "2026-01-04T12:57:04+00:00",
        "PasswordResetRequired": true
    }
}
[ec2-user@ip-10-0-10-102 ~]$
```

Create access keys for the user (save AccessKeyId and SecretAccessKey securely):

task6\_create\_access\_key\_output.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam create-access-key --user-name MyUserCli
{
    "AccessKey": {
        "UserName": "MyUserCli",
        "AccessKeyId": "XXXXXXXXXX",
        "Status": "Active",
        "SecretAccessKey": "XXXXXXXXXXXXXX",
        "CreateDate": "2026-01-04T12:59:34+00:00"
    }
}
[ec2-user@ip-10-0-10-102 ~]$
```

List access keys:

task6\_create\_access\_key\_output.png

```
[ec2-user@ip-10-0-10-102 ~]$ aws iam list-access-keys --user-name MyUserCli
{
    "AccessKeyMetadata": [
        {
            "UserName": "MyUserCli",
            "AccessKeyId": "XXXXXXXXXX",
            "Status": "Active",
            "CreateDate": "2026-01-04T12:59:34+00:00"
        }
    ]
}
[ec2-user@ip-10-0-10-102 ~]$
```

Use environment variables to authenticate as that user in the Codespace:

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

```
[ec2-user@ip-10-0-10-102 ~]$ export AWS_ACCESS_KEY_ID=
[ec2-user@ip-10-0-10-102 ~]$ export AWS_SECRET_ACCESS_KEY=
[ec2-user@ip-10-0-10-102 ~]$ printenv | grep AWS_
AWS_SECRET_ACCESS_KEY=
AWS_ACCESS_KEY_ID=
[ec2-user@ip-10-0-10-102 ~]$ aws iam get-user --user-name MyUserCli
An error occurred (AccessDenied) when calling the GetUser operation: User: arn:aws:iam::542622959327:user/MyUserCli is not authorized to perform: iam:GetUser on resource: user/MyUserCli because no identity-based policy allows the iam:GetUser action
[ec2-user@ip-10-0-10-102 ~]$ exit
logout
Connection to 3.29.21.250 closed.
@hamna-mahmood ✘ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

After clearing or switching credentials, repeat get-user and save:

task6\_after\_logout\_get\_user\_success.png

```
@hamna-mahmood ✘ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ unset AWS_ACCESS_KEY_ID
@hamna-mahmood ✘ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ unset AWS_SECRET_ACCESS_KEY
@hamna-mahmood ✘ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ printenv | grep AWS_
@hamna-mahmood ✘ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws iam get-user --user-name MyUserCli
{
    "User": {
        "Path": "/",
        "UserName": "MyUserCli",
        "UserId": "AIDAX4VWZI3PYBIHCD3HA",
        "Arn": "arn:aws:iam::542622959327:user/MyUserCli",
        "CreateDate": "2026-01-04T12:30:34+00:00"
    }
}
```

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

task7\_filter\_by\_tag\_public\_ip.png

```
@hamna-mahmood eworkspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-instances \
>   --filters "Name=tag:Name,Values=MyServer" \
>   --query "Reservations[*].Instances[*].PublicIpAddress" \
>   --output text
3.28.135.154
3.29.21.250
@hamna-mahmood eworkspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

task7\_filter\_by\_instance\_type.png

```
@hamna-mahmood eworkspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-instances \
>   --filters "Name=instance-type,Values=t3.micro" \
>   --query "Reservations[].Instances[.].InstanceId" \
>   --output table
| DescribeInstances |
+-----+
| i-0fa9d64afed5822d6 |
| i-01b0026b0394a0b56 |
| i-09dc2f89d7b6a07a5 |
| i-06a6d1c601e3e9e60 |
| i-01d026d26fc0ea152 |
| i-046929aa82fc20cce |
+-----+
...skipping...
| DescribeInstances |
+-----+
| i-0fa9d64afed5822d6 |
| i-01b0026b0394a0b56 |
| i-09dc2f89d7b6a07a5 |
| i-06a6d1c601e3e9e60 |
| i-01d026d26fc0ea152 |
| i-046929aa82fc20cce |
+-----+
(END)...skipping...
| DescribeInstances |
+-----+
| i-0fa9d64afed5822d6 |
| i-01b0026b0394a0b56 |
| i-09dc2f89d7b6a07a5 |
| i-06a6d1c601e3e9e60 |
| i-01d026d26fc0ea152 |
| i-046929aa82fc20cce |
+-----+
```

task7\_filter\_by\_subnet.png

```
@hamna-mahmood eworkspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-instances --filters "Name=subnet-id,Values=subnet-0a2cee75f0ad691a6" --query "Reservations[*].Instances[*].InstanceId" --output table
-----+
| DescribeInstances |
+-----+
| i-0fa9d64afed5822d6 |
| i-01b0026b0394a0b56 |
| i-09dc2f89d7b6a07a5 |
| i-06a6d1c601e3e9e60 |
| i-01d026d26fc0ea152 |
| i-046929aa82fc20cce |
+-----+
```

task7\_filter\_by\_vpc.png

```
@hamna-mahmood eworkspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-instances --filters "Name=vpc-id,Values=vpc-09822723aed1d2815" --query "Reservations[*].Instances[*].InstanceId" --output table
-----+
| DescribeInstances |
+-----+
| i-0fa9d64afed5822d6 |
| i-01b0026b0394a0b56 |
| i-09dc2f89d7b6a07a5 |
| i-06a6d1c601e3e9e60 |
| i-01d026d26fc0ea152 |
| i-046929aa82fc20cce |
+-----+
@hamna-mahmood eworkspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $
```

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

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

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (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-01d026d26fc0ea152 | 3.28.135.154 | MyServer |
| i-046929aa82fc20cce | 3.29.21.250 | MyServer |
+-----+-----+-----+
```

task8\_query\_table\_instance\_state.png

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-instances \
>   --query "Reservations[*].Instances[*].[InstanceId,State.Name]" \
>   --output table
-----
|           DescribeInstances           |
+-----+-----+
| i-0fa9d64afed5822d6 | running |
| i-01b0026b0394a0b56 | running |
| i-09dc2f89d7b6a07a5 | running |
| i-06a6d1c601e3e9e60 | running |
| i-01d026d26fc0ea152 | running |
| i-046929aa82fc20cce | running |
+-----+-----+
```

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

```
@hamna-mahmood ② /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 describe-instances \
>   --query "Reservations[*].Instances[*].[InstanceId,InstanceType,Placement.AvailabilityZone]" \
>   --output table
-----
|           DescribeInstances           |
+-----+-----+-----+
| i-0fa9d64afed5822d6 | t3.micro | me-central-1a |
| i-01b0026b0394a0b56 | t3.micro | me-central-1a |
| i-09dc2f89d7b6a07a5 | t3.micro | me-central-1a |
| i-06a6d1c601e3e9e60 | t3.micro | me-central-1a |
| i-01d026d26fc0ea152 | t3.micro | me-central-1a |
| i-046929aa82fc20cce | t3.micro | me-central-1a |
+-----+-----+-----+
```

## Cleanup — Remove resources to avoid charges

cleanup\_terminate\_instance.png

```
hamna-mahmood ✭ /workspaces/CC-Hamna-Mahmood-25-BSE-VA (main) $ aws ec2 terminate-instances --instance-ids i-0fa9d64afed5822d6
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-0fa9d64afed5822d6",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

cleanup\_delete\_volumes\_snapshots.png

Successfully deleted 2 volumes

Volumes Info

Saved Filter sets: Choose filter set ▾

Last updated less than a minute ago

Recycle Bin Actions Create volume

Name | Volume ID | Type | Size | IOPS | Throughput | Snapshot ID | Source volume ID | Created | Availability Zone

You currently have no volumes in this region

Fault tolerance for all volumes in this Region

Snapshot summary

Recently backed up volumes / Total # volumes

0 / 2

Data Lifecycle Manager default policy for EBS Snapshots status

No default policy set up | Create policy

Last updated on Mon, Jan 05, 2026, 03:35:17 PM (GMT+05:00)

Snapshots Info

Owned by me ▾

Last updated less than a minute ago

Recycle Bin Actions Create snapshot

Name | Snapshot ID | Full snapshot size | Volume size | Description | Storage tier | Snapshot status | Started | Progress

You currently have no snapshots in this Region.

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

These groups cannot be deleted as they are default.

The screenshot shows two CloudFormation stacks. The top stack, titled 'Security Groups (2)', lists two default security groups: 'sg-09b8b587e97008371' and 'sg-03a28557d7dc9c211'. The bottom stack, titled 'Key pairs (0)', shows a successful deletion of 5 key pairs. A green banner at the top of the bottom stack indicates 'Successfully deleted 5 key pairs'.

cleanup\_iam\_users\_deleted.png

The screenshot shows the IAM 'Users (0)' page. It displays a search bar and a table header with columns: User name, Path, Group:, Last activity, MFA, and Password age. Below the header, a message says 'No resources to display'.

cleanup\_summary.png

Only default security groups remain.

The screenshot displays several CloudFormation stacks. On the left, the 'User groups (0)' stack shows a table of terminated user groups: prod-web-3-3, prod-web-1-1, prod-nginx-pr..., prod-web-2-2, MyServer, and MyServer. The 'Security Groups (2)' stack shows the same two default security groups listed above. The 'Volumes' section shows no volumes. The 'Snapshot summary' section shows '0 / 0' snapshots.

Key pairs		Info
<input type="text"/> Find Key Pair by attribute or tag		
Name	Type	Created
No key pairs to display		