AWS CLI — Practical Tasks Completed

Complete step-by-step AWS CLI commands for tasks you completed. Replace placeholder names with your actual resource names before running.

Author: Samarth Mahadik

Tasks included (TOC):

1) Create IAM group, user, attach policy

Purpose: Create an IAM group and user, attach a managed policy (or custom policy) to the group, then add the user into the group.

```
Commands (replace names where indicated):

# Create group aws iam create-
group --group-name DevGroup

# Create user aws iam
create-user --user-name
devuser

# (Optional) Create access keys for user (store
these securely) aws iam create-access-key --
user-name devuser

# Create or attach a policy to the group (example: AmazonS3FullAccess) aws
iam attach-group-policy --group-name DevGroup --policy-arn
arn:aws:iam::aws:policy/AmazonS3FullAccess

# Add the user to the group aws iam add-user-to-
group --user-name devuser --group-name DevGroup

# Verify group members and
attached policies aws iam get-
group --group-name DevGroup
aws iam list-attached-group-policies --group-name DevGroup
```

```
   ~~~        /
     ~~._.   _/
        _/ _/
       _/m/'
Last login: Wed Sep  3 06:01:00 2025 from 18.206.107.27
[ec2-user@ip-172-31-30-203 ~]$ ^[[200~aws iam create-group --group-name DevTeam
-bash: $'\E[200~aws': command not found
[ec2-user@ip-172-31-30-203 ~]$ aws iam create-group --group-name DevTeam
{
    "Group": {
        "Path": "/",
        "GroupName": "DevTeam",
        "GroupId": "AGPAWJCBWYZDIW3QUFKSY",
        "Arn": "arn:aws:iam::431782020678:group/DevTeam",
        "CreateDate": "2025-09-04T02:25:01+00:00"
    }
}
[ec2-user@ip-172-31-30-203 ~]$ aws iam create-user --user-name samarth
{
    "User": {
        "Path": "/",
        "UserName": "samarth",
        "UserId": "AIDAWJCBWYZDDHGEWCOWM",
        "Arn": "arn:aws:iam::431782020678:user/samarth",
        "CreateDate": "2025-09-04T02:25:57+00:00"
    }
}
[ec2-user@ip-172-31-30-203 ~]$ aws iam add-user-to-group --user-name samarth --group-name DevTeam
[ec2-user@ip-172-31-30-203 ~]$ aws iam attach-group-policy \
    --group-name DevTeam \
    --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
[ec2-user@ip-172-31-30-203 ~]$
```

2) Create S3 Bucket and Upload File

Purpose: Create a versioned S3 bucket and upload a file using AWS CLI.

Commands:

```
# Create bucket (example for ap-south-1 region) aws s3api create-bucket --
bucket my-src-bucket --region ap-south-1 --create-bucket-configuration
LocationCon

# For us-east-1 (N. Virginia) omit the LocationConstraint flag:
# aws s3api create-bucket --bucket my-dst-bucket-prod --region us-east-1

# Enable versioning on the bucket aws s3api put-bucket-versioning --
bucket my-src-bucket --versioning-configuration Status=Enabled

# Upload file aws s3 cp ./local-file.txt
s3://my-src-bucket/path/local-file.txt

# List objects to verify aws
s3 ls s3://my-src-bucket -
recursive
```

aws Search [Alt+S] United States (N. Virginia) Account ID: 4317-8202-0678 Samarth Mahadik

```
{
    "Group": {
        "Path": "/",
        "GroupName": "DevTeam",
        "GroupId": "AGPAWJCBWYZDIW3QUFKSY",
        "Arn": "arn:aws:iam::431782020678:group/DevTeam",
        "CreateDate": "2025-09-04T02:25:01+00:00"
    }
}
[ec2-user@ip-172-31-30-203 ~]$ aws iam create-user --user-name samarth
{
    "User": {
        "Path": "/",
        "UserName": "samarth",
        "UserId": "AIDAWJCBWYZDDHGEWCOWM",
        "Arn": "arn:aws:iam::431782020678:user/samarth",
        "CreateDate": "2025-09-04T02:25:57+00:00"
    }
}
[ec2-user@ip-172-31-30-203 ~]$ aws iam add-user-to-group --user-name samarth --group-name DevTeam
[ec2-user@ip-172-31-30-203 ~]$ aws iam attach-group-policy \
    --group-name DevTeam \
    --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
[ec2-user@ip-172-31-30-203 ~]$ aws s3 mb s3://my-unique-bucket-name-123 --region us-east-1
make_bucket failed: s3://my-unique-bucket-name-123 An error occurred (BucketAlreadyExists) when calling the CreateBucket operation: The requested bucket name is not a
vailable. The bucket namespace is shared by all users of the system. Please select a different name and try again.
[ec2-user@ip-172-31-30-203 ~]$ aws s3 mb s3://samarth-bucket-name-07 --region us-east-1
make_bucket: samarth-bucket-name-07
[ec2-user@ip-172-31-30-203 ~]$ echo "This is my first S3 file upload via AWS CLI" > sample.txt
[ec2-user@ip-172-31-30-203 ~]$ aws s3 cp sample.txt s3://samarth-bucket-name-07/
upload: ./sample.txt to s3://samarth-bucket-name-07/sample.txt
[ec2-user@ip-172-31-30-203 ~]$
```

CloudShell Feedback

 Privacy Terms Cookie preferences

3) Create key pair and launch EC2 instance

Purpose: Create an EC2 key pair locally and launch an instance using that key pair.

```
Commands:

# Create a key pair and save the private key (example) aws ec2
create-key-pair --key-name MyKeyPair --query 'KeyMaterial' --output
text > MyKeyPair.pem chmod 400 MyKeyPair.pem

# Create (or reuse) a security group that allows SSH (port 22) # If using
default VPC, you can create a security group like below (replace VPC ID)
aws ec2 create-security-group --group-name ssh-sg --description "SSH
access" --vpc-id <vpc-id> aws ec2 authorize-security-group-ingress --
group-id <sg-id> --protocol tcp --port 22 --cidr 0.0.0.0/0

# Launch an instance (replace AMI, subnet, security-group-id, key-name) aws
ec2 run-instances --image-id ami-0123456789abcdef0 --count 1 --instance-type
t2.micro   --key-name MyKey

# Get the instance id aws ec2 describe-instances --filters "Name=key-
name,Values=MyKeyPair" --query "Reservations[*].Instances[*].
```

aws Search [Alt+S] United States (N. Virginia) Account ID: 4317-8202-0678 Samarth Mahadik

```
        "GroupName": "DevTeam",
        "GroupId": "AGPAWJCBWYZDIW3QUFKSY",
        "Arn": "arn:aws:iam::431782020678:group/DevTeam",
        "CreateDate": "2025-09-04T02:25:01+00:00"
    }
}
[ec2-user@ip-172-31-30-203 ~]$ aws iam create-user --user-name samarth
{
    "User": {
        "Path": "/",
        "UserName": "samarth",
        "UserId": "AIDAWJCBWYZDDHGEWCOWM",
        "Arn": "arn:aws:iam::431782020678:user/samarth",
        "CreateDate": "2025-09-04T02:25:57+00:00"
    }
}
[ec2-user@ip-172-31-30-203 ~]$ aws iam add-user-to-group --user-name samarth --group-name DevTeam
[ec2-user@ip-172-31-30-203 ~]$ aws iam attach-group-policy \
    --group-name DevTeam \
    --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
[ec2-user@ip-172-31-30-203 ~]$ aws s3 mb s3://my-unique-bucket-name-123 --region us-east-1
make_bucket failed: s3://my-unique-bucket-name-123 An error occurred (BucketAlreadyExists) when calling the CreateBucket operation: The requested bucket name is not a
vailable. The bucket namespace is shared by all users of the system. Please select a different name and try again.
[ec2-user@ip-172-31-30-203 ~]$ aws s3 mb s3://samarth-bucket-name-07 --region us-east-1
make_bucket: samarth-bucket-name-07
[ec2-user@ip-172-31-30-203 ~]$ echo "This is my first S3 file upload via AWS CLI" > sample.txt
[ec2-user@ip-172-31-30-203 ~]$ aws s3 cp sample.txt s3://samarth-bucket-name-07/
upload: ./sample.txt to s3://samarth-bucket-name-07/sample.txt
[ec2-user@ip-172-31-30-203 ~]$ aws ec2 create-key-pair \
    --key-name MyKeyPair \
    --query 'KeyMaterial' \
    --output text > MyKeyPair.pem
[ec2-user@ip-172-31-30-203 ~]$ chmod 400 MyKeyPair.pem
```

CloudShell Feedback

 Privacy Terms Cookie preferences

4) Stop and Terminate EC2 Instance

Purpose: Stop a running EC2 instance and then terminate it when testing is done.

```
Commands:

# Stop instance aws ec2 stop-instances --
instance-ids i-0abcdef1234567890

# Wait until stopped (optional) aws ec2 wait
instance-stopped --instance-ids i-
0abcdef1234567890

# Terminate instance aws ec2 terminate-
instances --instance-ids i-0abcdef1234567890

# Wait until terminated (optional) aws ec2 wait
instance-terminated --instance-ids i-
0abcdef1234567890

# Verify state aws ec2 describe-instances --instance-ids i-0abcdef1234567890 -
-query "Reservations[*].Instances[*].[Instanc
```

aws Search [Alt+S] United States (N. Virginia) Account ID: 4317-8202-0678 Samarth Mahadik

```
[ec2-user@ip-172-31-30-203 ~]$ aws ec2 stop-instances --instance-ids i-0fe52eeb3d501e559
{
    "StoppingInstances": [
        {
            "InstanceId": "i-0fe52eeb3d501e559",
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
[ec2-user@ip-172-31-30-203 ~]$ aws ec2  terminate-instances --instance-ids i-0fe52eeb3d501e559
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-0fe52eeb3d501e559",
            "CurrentState": {
                "Code": 48,
                "Name": "terminated"
            },
            "PreviousState": {
                "Code": 80,
                "Name": "stopped"
            }
        }
    ]
}
```

CloudShell Feedback

Privacy Terms Cookie preferences

5) Create and Attach IAM Role to EC2 (S3 access)

Purpose: Create an IAM role that EC2 can assume and grant it S3 permissions via an attached policy.

```
Steps & Commands:

# 1) Create trust policy file (trust-policy.json) with content:
# {
#   "Version": "2012-10-17",
#   "Statement": [
#     {
#       "Effect": "Allow",
#       "Principal": { "Service": "ec2.amazonaws.com" },
#       "Action": "sts:AssumeRole"
#     }
#   ]
# }

# 2) Create the role aws iam create-role --role-name EC2S3Role --
assume-role-policy-document file://trust-policy.json

# 3) Attach managed policy (example: AmazonS3ReadOnlyAccess) or a custom
policy aws iam attach-role-policy --role-name EC2S3Role --policy-arn
arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess

# 4) Create instance profile and add role to profile aws iam
create-instance-profile --instance-profile-name EC2S3Role aws iam
add-role-to-instance-profile --instance-profile-name EC2S3Role --
role-name EC2S3Role

# 5) Attach the instance profile to an existing instance
aws ec2 associate-iam-instance-profile --instance-id i-0abcdef1234567890 --
iam-instance-profile Name=EC2S3Ro

# 6) Verify from instance (once attached) you can access S3
(no need for access keys): # From inside EC2 (with AWS CLI
configured via instance profile): aws s3 ls s3://my-src-bucket
```

aws Search [Alt+S] United States (N. Virginia) Account ID: 4317-8202-0678 Samarth Mahadik

```
[ec2-user@ip-172-31-30-203 ~]$ nano trust-policy.json
[ec2-user@ip-172-31-30-203 ~]$ aws iam create-role     --role-name EC2-S3-Access-Role     --assume-role-policy-document file://trust-policy.json
{
    "Role": {
        "Path": "/",
        "RoleName": "EC2-S3-Access-Role",
        "RoleId": "AROAWJCBWYZDE45QQYUHO",
        "Arn": "arn:aws:iam::431782020678:role/EC2-S3-Access-Role",
        "CreateDate": "2025-09-04T03:12:33+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "ec2.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
[ec2-user@ip-172-31-30-203 ~]$ aws iam attach-role-policy \
    --role-name EC2-S3-Access-Role \
    --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
[ec2-user@ip-172-31-30-203 ~]$ aws iam create-instance-profile \
    --instance-profile-name EC2-S3-Instance-Profile
{
    "InstanceProfile": {
        "Path": "/",
        "InstanceProfileName": "EC2-S3-Instance-Profile",
```

CloudShell Feedback

 Privacy Terms Cookie preferences

aws Search [Alt+S] United States (N. Virginia) Account ID: 4317-8202-0678 Samarth Mahadik

```
        }
    }
}
[ec2-user@ip-172-31-30-203 ~]$ aws iam attach-role-policy \
    --role-name EC2-S3-Access-Role \
    --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
[ec2-user@ip-172-31-30-203 ~]$ aws iam create-instance-profile \
    --instance-profile-name EC2-S3-Instance-Profile
{
    "InstanceProfile": {
        "Path": "/",
        "InstanceProfileName": "EC2-S3-Instance-Profile",
        "InstanceProfileId": "AIPAWJCBWYZDIKTI7P5I6",
        "Arn": "arn:aws:iam::431782020678:instance-profile/EC2-S3-Instance-Profile",
        "CreateDate": "2025-09-04T03:13:57+00:00",
        "Roles": []
    }
}
[ec2-user@ip-172-31-30-203 ~]$ aws ec2 associate-iam-instance-profile \
    --instance-id i-08c30b8c5ecd02f85 \
    --iam-instance-profile Name=EC2-S3-Instance-Profile
{
    "IamInstanceProfileAssociation": {
        "AssociationId": "iip-assoc-0e03a5bc48fc71814",
        "InstanceId": "i-08c30b8c5ecd02f85",
        "IamInstanceProfile": {
            "Arn": "arn:aws:iam::431782020678:instance-profile/EC2-S3-Instance-Profile",
            "Id": "AIPAWJCBWYZDIKTI7P5I6"
        },
        "State": "associating"
    }
}
```

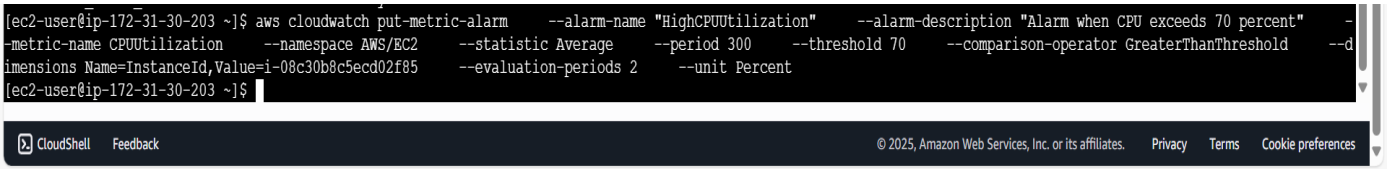
CloudShell Feedback Privacy Terms Cookie preferences

6) Create CloudWatch alarm for CPU utilization

Purpose: Trigger notification (SNS) when an EC2 instance CPU crosses the threshold (example: ≥80%).

```
Command (replace instance-id and SNS topic ARN):

aws cloudwatch put-metric-alarm --alarm-name HighCPUAlarm --alarm-description
"CPU >= 80% for 1 datapoint (5   # Verify alarm
aws cloudwatch describe-alarms --alarm-names HighCPUAlarm
```



7) Create SNS topic, Subscribe SQS, and Publish message

Purpose: Create an SNS topic, create an SQS queue, subscribe the queue to the topic, and publish 'Hello World' message.

```
Commands & steps:

# 1) Create SNS topic aws sns create-topic --name highcpu-
topic # Note the TopicArn returned (e.g., arn:aws:sns:us-
east-1:123456789012:highcpu-topic)

# 2) Create SQS queue aws sqs create-queue --queue-name highcpu-queue # Get
the queue URL and attributes (to retrieve ARN) aws sqs get-queue-attributes --
queue-url https://sqs.us-east-1.amazonaws.com/123456789012/highcpu-queue --at

# 3) Allow SNS to send messages to SQS by setting queue
policy (queue-policy.json) # Example minimal policy:
# {
#   "Version":"2012-10-17",
#   "Id":"Allow-SNS-SendMessage",
#   "Statement":[
#     {
#       "Effect":"Allow",
#       "Principal":"*",
#       "Action":"sqs:SendMessage",
#       "Resource":"arn:aws:sqs:us-east-1:123456789012:highcpu-queue",
#       "Condition":{"ArnEquals":{"aws:SourceArn":"arn:aws:sns:us-east-
1:123456789012:highcpu-topic"}}
#     }
#   ] # } aws sqs set-queue-attributes --queue-url <QueueUrl> --
attributes file://queue-policy.json

# 4) Subscribe SQS to SNS aws sns subscribe --topic-arn arn:aws:sns:us-east-
1:123456789012:highcpu-topic --protocol sqs --notification

# 5) Publish a message aws sns publish --topic-arn arn:aws:sns:us-east-
1:123456789012:highcpu-topic --message "Hello World"

# 6) Read message from SQS (to verify)
aws sqs receive-message --queue-url https://sqs.us-east-
1.amazonaws.com/123456789012/highcpu-queue --max-num
```

```
[ec2-user@ip-172-31-30-203 ~]$ aws sns create-topic --name MyDemoTopic
{
    "TopicArn": "arn:aws:sns:us-east-1:431782020678:MyDemoTopic"
}
[ec2-user@ip-172-31-30-203 ~]$ aws sqs create-queue --queue-name MyDemoQueue
{
    "QueueUrl": "https://sqs.us-east-1.amazonaws.com/431782020678/MyDemoQueue"
}
[ec2-user@ip-172-31-30-203 ~]$ aws sns subscribe \
    --topic-arn arn:aws:sns:us-east-1:431782020678:MyDemoTopic^C
    --protocol sqs \
    --notification-endpoint <QueueArn>

[ec2-user@ip-172-31-30-203 ~]$ aws sqs set-queue-attributes \
  --queue-url https://sqs.us-east-1.amazonaws.com/431782020678/MyDemoQueue \
  --attributes '{
    "Policy":"{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Principal\":{\"Service\":\"sns.amazonaws.com\"},\"Action\":\"sqs:SendMessage\",\"Reso
urce\"}]}}"
  }'
```

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

8) Create VPC, Subnet, and Internet Gateway

Purpose: Create basic networking components (VPC + public subnet + IGW + route) to host public instances.

```
Commands:

# Create VPC aws ec2 create-vpc
--cidr-block 10.0.0.0/16 # Note
VpcId returned (e.g., vpc-
0abcd1234)

# Create subnet aws ec2 create-subnet --vpc-id vpc-0abcd1234 --cidr-
block 10.0.1.0/24 --availability-zone us-east-1a # Note SubnetId
returned

# Create Internet Gateway and attach to VPC aws ec2 create-
internet-gateway aws ec2 attach-internet-gateway --internet-
gateway-id igw-0abcd1234 --vpc-id vpc-0abcd1234
```

aws Search [Alt+S] United States (N. Virginia) Account ID: 4317-8202-0678 Samarth Mahadik

```
        "Ipv6Native": false,
        "PrivateDnsNameOptionsOnLaunch": {
            "HostnameType": "ip-name",
            "EnableResourceNameDnsARecord": false,
            "EnableResourceNameDnsAAAARecord": false
        },
        "SubnetId": "subnet-0ad284648d13fbd94",
        "State": "available",
        "VpcId": "vpc-042010a8202a1caf9",
        "CidrBlock": "10.0.1.0/24",
        "AvailableIpAddressCount": 251,
[ec2-user@ip-172-31-30-203 ~]$ aws ec2 create-internet-gateway \
    --tag-specifications 'ResourceType=internet-gateway,Tags=[{Key=Name,Value=MyIGW}]'
{
    "InternetGateway": {
        "Attachments": [],
        "InternetGatewayId": "igw-0223d0915d8f80a15",
        "OwnerId": "431782020678",
        "Tags": [
            {
                "Key": "Name",
                "Value": "MyIGW"
            }
        ]
    }
}
[ec2-user@ip-172-31-30-203 ~]$
```

i-08c30b8c5ecd02f85 (Linux server)

PublicIPs: 50.17.33.61 PrivateIPs: 172.31.30.203

CloudShell Feedback

Privacy Terms Cookie preferences

aws Search [Alt+S] United States (N. Virginia) Account ID: 4317-8202-0678 Samarth Mahadik

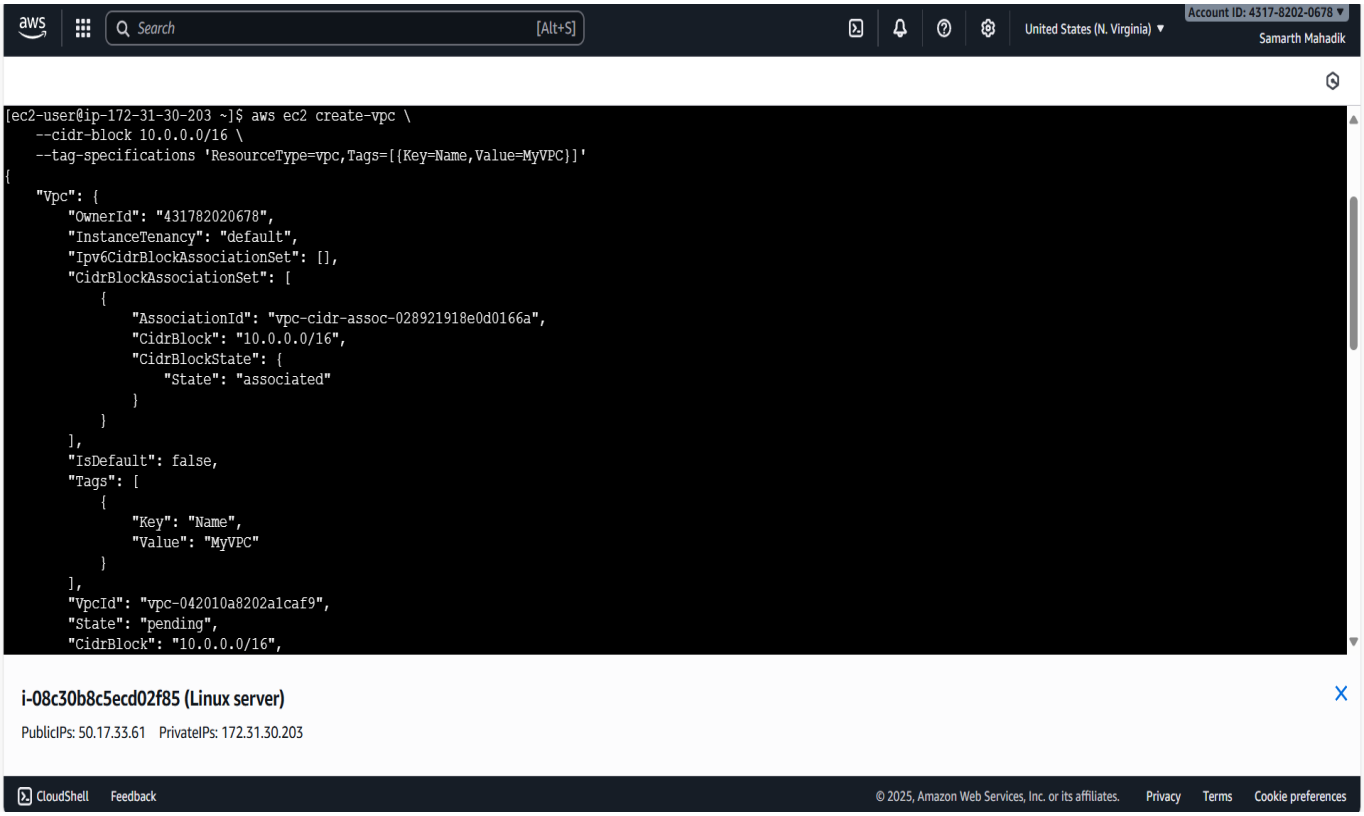
```
[ec2-user@ip-172-31-30-203 ~]$ aws ec2 create-subnet \
    --vpc-id vpc-042010a8202a1caf9 \
    --cidr-block 10.0.1.0/24 \
    --availability-zone us-east-1a \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=MyPublicSubnet}]'
{
    "Subnet": {
        "AvailabilityZoneId": "use1-az2",
        "MapCustomerOwnedIpOnLaunch": false,
        "OwnerId": "431782020678",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [],
        "Tags": [
            {
                "Key": "Name",
                "Value": "MyPublicSubnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-east-1:431782020678:subnet/subnet-0ad284648d13fbd94",
        "EnableDns64": false,
        "Ipv6Native": false,
        "PrivateDnsNameOptionsOnLaunch": {
            "HostnameType": "ip-name",
            "EnableResourceNameDnsARecord": false,
            "EnableResourceNameDnsAAAARecord": false
        },
        "SubnetId": "subnet-0ad284648d13fbd94",
```

i-08c30b8c5ecd02f85 (Linux server)

PublicIPs: 50.17.33.61 PrivateIPs: 172.31.30.203

CloudShell Feedback

 Privacy Terms Cookie preferences



Verification & Cleanup

After completing tests, verify resources and clean them to avoid charges. Useful commands:

```
# List / verify resources aws ec2 describe-instances --filters "Name=instance-state-name,Values=running" aws s3 ls aws iam list-users aws sns list-topics aws sqs list-queues aws ec2 describe-vpcs

# Cleanup examples (replace IDs) aws s3 rm s3://my-src-bucket --recursive aws s3api delete-bucket --bucket my-src-bucket aws ec2 terminate-instances --instance-ids i-0abcdef1234567890 aws iam remove-user-from-group --user-name devuser --group-name DevGroup aws iam delete-user --user-name devuser aws iam delete-group --group-name
```

```
DevGroup aws sns delete-topic --topic-arn
arn:aws:sns:... aws sqs delete-queue --queue-url
https://sqs... aws ec2 detach-internet-gateway --
internet-gateway-id igw-... --vpc-id vpc-... aws ec2
delete-internet-gateway --internet-gateway-id igw-...
aws ec2 delete-subnet --subnet-id subnet-... aws ec2
delete-vpc --vpc-id vpc-...
```

Final Notes

• Replace placeholder names and ARNs with your real resource identifiers before running commands. • Use least-privilege principles for IAM policies. Avoid wide open policies in production. • Keep private keys secure (chmod 400) and never commit them to source control. • Test carefully: some operations (terminate/delete) are destructive.