

## **Command Line Interface Lab 1 Report**

Samuel Kojo Andam Quansah

University of Maryland Global Campus

CCS 625: Network Engineering- Project 1

Instructor Name: Dr. Lipton

May 13,2025

## Introduction

This lab exercise focused on using the AWS Command Line Interface (CLI) via AWS CloudShell to provision cloud infrastructure resources. The objective was to demonstrate foundational CLI skills in AWS by creating a Virtual Private Cloud (VPC), provisioning and tagging an Internet Gateway, and attaching that gateway to the VPC. These are fundamental steps in configuring public access to resources within a VPC and are essential for understanding networking within AWS environments.

## Step-by-Step Methodology

### Step 1: Access the AWS CloudShell Service

I logged into the UMGC Virtual Lab Environment and selected the appropriate AWS StudentAdminAccess role. After launching the AWS Management Console, I navigated to the CloudShell service and dismissed the welcome prompt. I explored available CloudShell features to prepare the terminal for use.

### Step 2: Explore AWS CLI Commands

Used the following command to explore the help system:

```
aws help
```

Verified the command structure:

```
aws <service> <subcommand> [--options]
```

Navigated CLI manual pages using arrow keys and space bar for full documentation access.

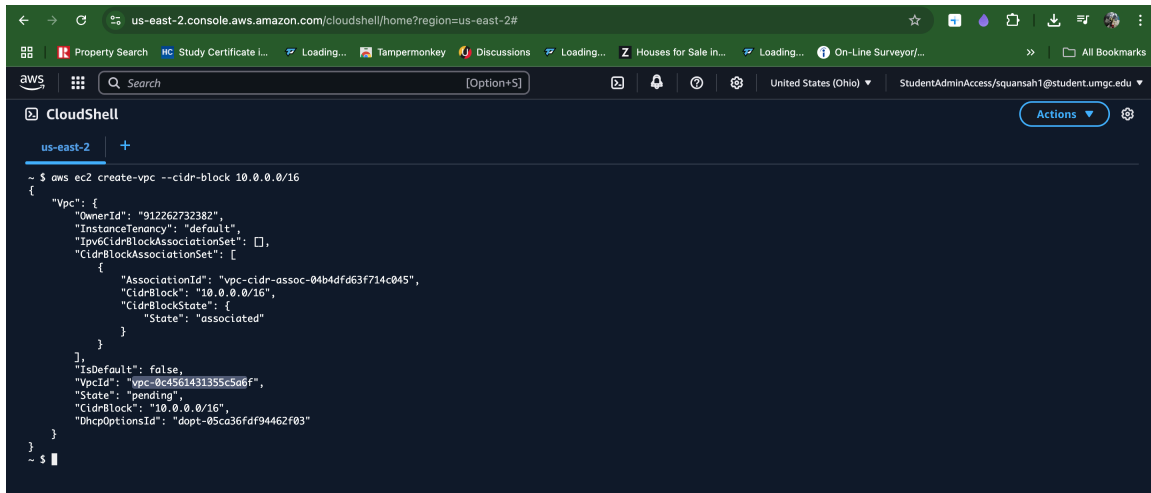
### Step 3: Provision a Virtual Private Cloud (VPC)

Created a new VPC using the following command:

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16
```

Output was in JSON format, and I noted the VPC ID: vpc-0c4561431355c5a6f

Screenshot 1: Output showing created VPC ID and details



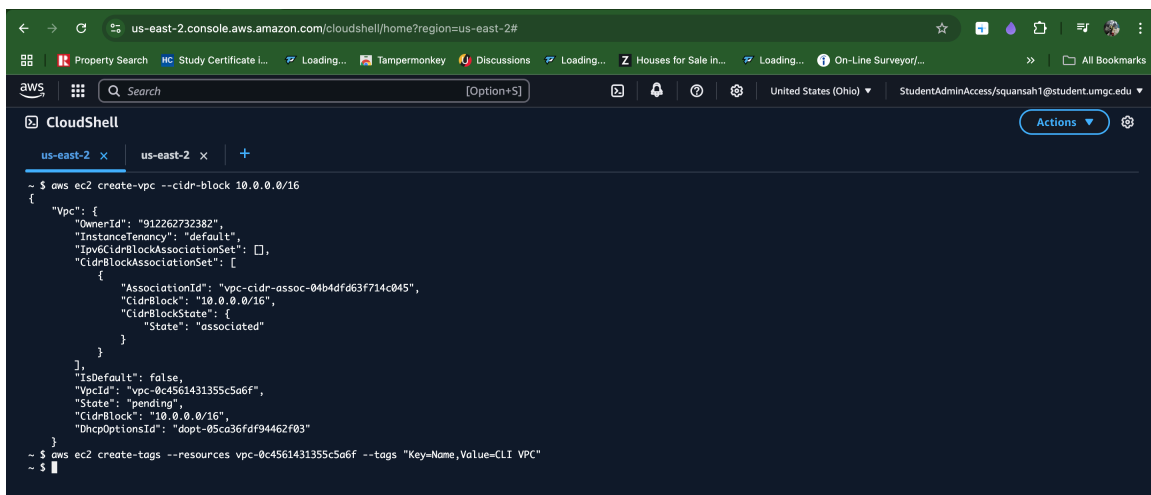
```
us-east-2 console.aws.amazon.com/cloudshell/home?region=us-east-2#
Property Search Study Certificate L... Loading... Tampermonkey Discussions Loading... Z Houses for Sale in... Loading... On-Line Surveyor/...
AWS Search [Option+S] United States (Ohio) StudentAdminAccess/squansah1@student.umgc.edu
CloudShell Actions
us-east-2 +
~ $ aws ec2 create-vpc --cidr-block 10.0.0.0/16
{
  "Vpc": {
    "OwnerId": "912262732382",
    "InstanceTenancy": "default",
    "Ipv6CidrBlockAssociationSet": [],
    "CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-04b4dfd63f714c045",
        "CidrBlock": "10.0.0.0/16",
        "CidrBlockState": {
          "State": "associated"
        }
      }
    ],
    "IsDefault": false,
    "VpcId": "vpc-0c4561431355c5a6f",
    "State": "pending",
    "CidrBlock": "10.0.0.0/16",
    "DhcpOptionsId": "dopt-05ca36dfd94462f03"
  }
}
~ $
```

Figure 1 Output showing created VPC ID and details

Tagged the VPC for easy identification:

```
aws ec2 create-tags --resources vpc-0c4561431355c5a6f --tags Key=Name,Value="CLI VPC"
```

Screenshot 2: Command showing successful tag application (no output)



```
us-east-2 console.aws.amazon.com/cloudshell/home?region=us-east-2#
Property Search Study Certificate L... Loading... Tampermonkey Discussions Loading... Z Houses for Sale in... Loading... On-Line Surveyor/...
AWS Search [Option+S] United States (Ohio) StudentAdminAccess/squansah1@student.umgc.edu
CloudShell Actions
us-east-2 x us-east-2 x +
~ $ aws ec2 create-vpc --cidr-block 10.0.0.0/16
{
  "Vpc": {
    "OwnerId": "912262732382",
    "InstanceTenancy": "default",
    "Ipv6CidrBlockAssociationSet": [],
    "CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-04b4dfd63f714c045",
        "CidrBlock": "10.0.0.0/16",
        "CidrBlockState": {
          "State": "associated"
        }
      }
    ],
    "IsDefault": false,
    "VpcId": "vpc-0c4561431355c5a6f",
    "State": "pending",
    "CidrBlock": "10.0.0.0/16",
    "DhcpOptionsId": "dopt-05ca36dfd94462f03"
  }
}
~ $ aws ec2 create-tags --resources vpc-0c4561431355c5a6f --tags "Key=Name,Value=CLI VPC"
~ $
```

Figure 2 Command showing successful tag application (no output)

Verified the VPC creation in the AWS Console under Networking > VPCs

Screenshot 3: VPC Dashboard showing 'CLI VPC'

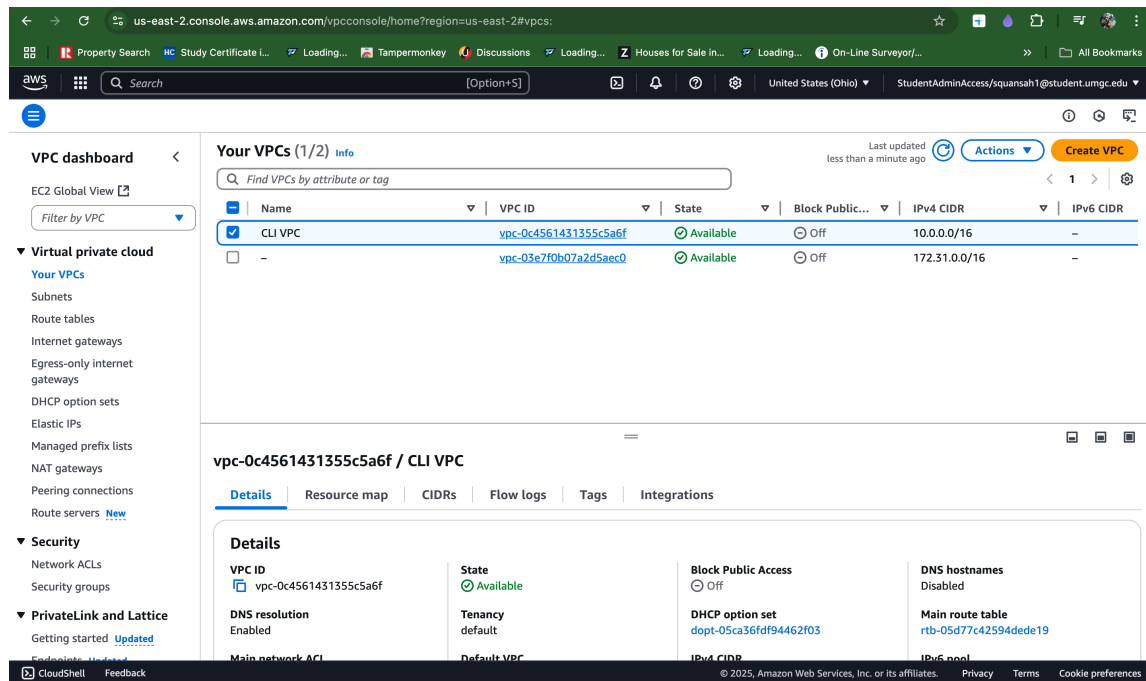


Figure 3 VPC Dashboard showing 'CLI VPC'

## Step 4: Provision and Attach an Internet Gateway

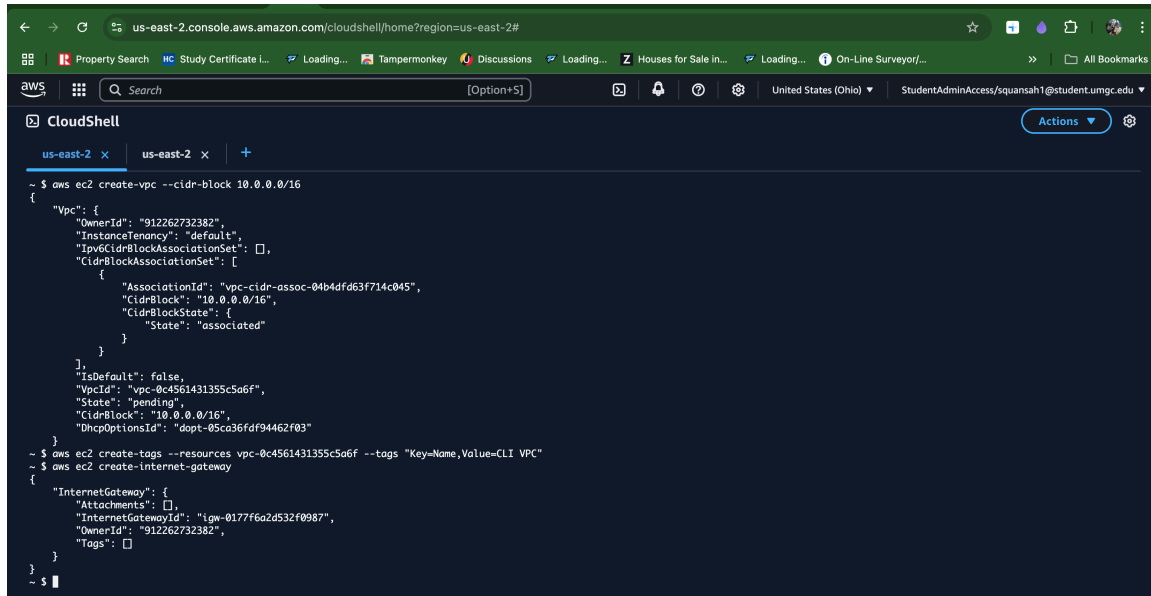
### 4.1 Create the Internet Gateway

Issued the following command:

```
aws ec2 create-internet-gateway
```

Noted the Internet Gateway ID: igw-0177f6a2d532f0987

#### Screenshot 4: Output showing created Internet Gateway ID



```
~ $ aws ec2 create-vpc --cidr-block 10.0.0.0/16
{
  "Vpc": {
    "OwnerId": "912262732382",
    "InstanceTenancy": "default",
    "Ipv6CidrBlockAssociationSet": [],
    "CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-04b4df63f714c045",
        "CidrBlock": "10.0.0.0/16",
        "CidrBlockState": {
          "State": "associated"
        }
      }
    ],
    "IsDefault": false,
    "VpcId": "vpc-0c4561431355c5a6f",
    "State": "pending",
    "CidrBlock": "10.0.0.0/16",
    "DhcpOptionsId": "dopt-05ca36fdf94462f03"
  }
}
~ $ aws ec2 create-tags --resources vpc-0c4561431355c5a6f --tags "Key=Name,Value=CLI VPC"
~ $ aws ec2 create-internet-gateway
{
  "InternetGateway": {
    "Attachments": [],
    "InternetGatewayId": "igw-0177f6a2d532f0987",
    "OwnerId": "912262732382",
    "Tags": []
  }
}
~ $
```

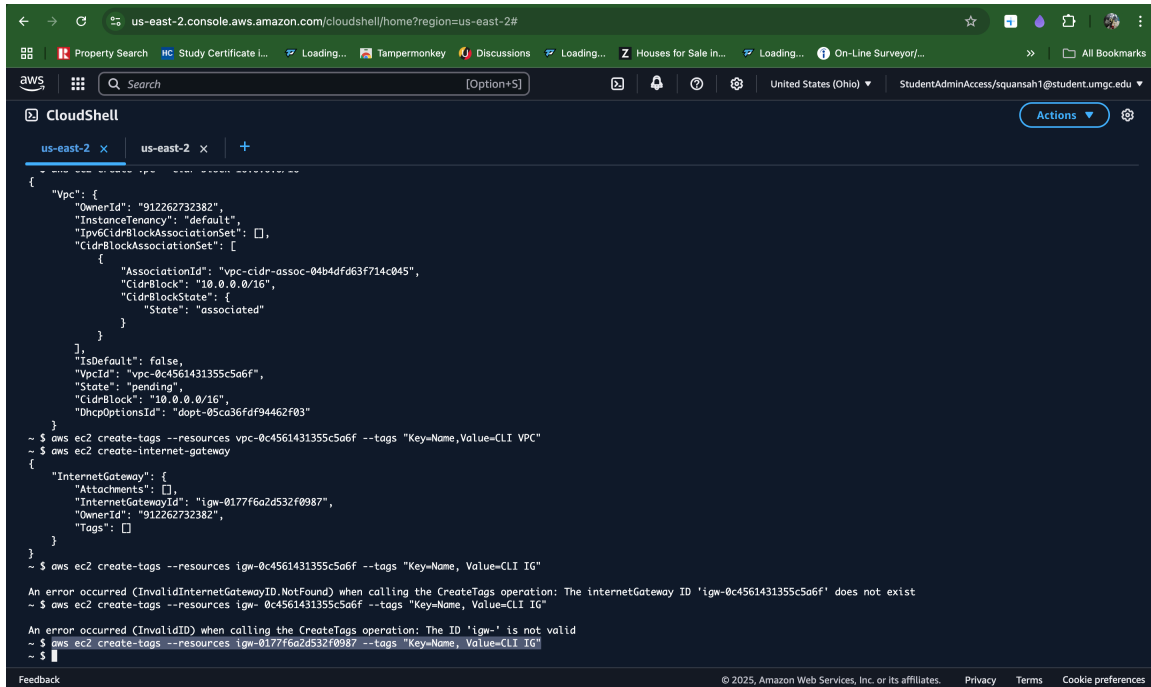
Figure 4 Output showing created Internet Gateway ID

#### 4.2 Tag the Internet Gateway

Tagged the gateway:

```
aws ec2 create-tags --resources igw-0177f6a2d532f0987 --tags Key=Name,Value="CLI IG"
```

## Screenshot 5: Tag applied successfully (no output)

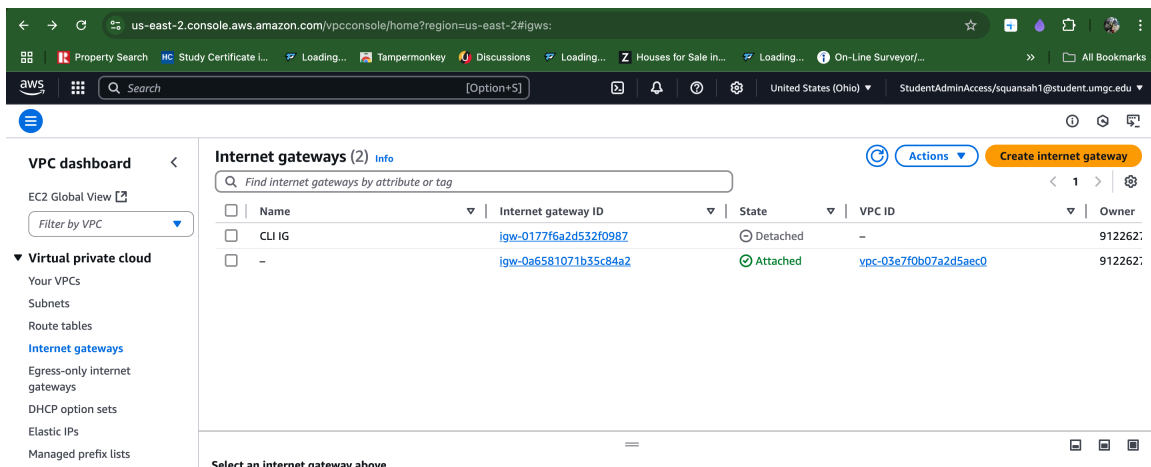


```
us-east-2 x us-east-2 x +
{
  "Vpc": {
    "OwnerId": "912262732382",
    "InstanceTenancy": "default",
    "Ipv6CidrBlockAssociationSet": [],
    "CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-04b4dfd63f714c045",
        "CidrBlock": "10.0.0.0/16",
        "CidrBlockState": {
          "State": "associated"
        }
      }
    ],
    "IsDefault": false,
    "VpcId": "vpc-0c4561431355c5a6f",
    "State": "pending",
    "CidrBlock": "10.0.0.0/16",
    "DhcpOptionsId": "dopt-05ca36fd94462f03"
  }
}
~ $ aws ec2 create-tags --resources vpc-0c4561431355c5a6f --tags "Key=Name,Value=CLI VPC"
~ $ aws ec2 create-internet-gateway
{
  "InternetGateway": {
    "Attachments": [],
    "InternetGatewayId": "igw-0177f6a2d532f0987",
    "OwnerId": "912262732382",
    "Tags": []
  }
}
~ $ aws ec2 create-tags --resources igw-0c4561431355c5a6f --tags "Key=Name, Value=CLI IG"
An error occurred (InvalidInternetGatewayID.NotFound) when calling the CreateTags operation: The internetGateway ID 'igw-0c4561431355c5a6f' does not exist
~ $ aws ec2 create-tags --resources igw-0c4561431355c5a6f --tags "Key=Name, Value=CLI IG"
An error occurred (InvalidID) when calling the CreateTags operation: The ID 'igw-' is not valid
~ $ aws ec2 create-tags --resources igw-0177f6a2d532f0987 --tags "Key=Name, Value=CLI IG"
~ $
```

Figure 5 Tag applied successfully (no output)

## Verified creation in AWS Console > Internet Gateways

### Screenshot 6: Internet Gateway in 'Detached' state



Internet gateways (2) Info					
Find internet gateways by attribute or tag					
<input type="checkbox"/>	Name	Internet gateway ID	State	VPC ID	Owner
<input type="checkbox"/>	CLI IG	igw-0177f6a2d532f0987	Detached	-	912262732382
<input type="checkbox"/>	-	igw-0a6581071b35c84a2	Attached	vpc-03e7f0b07a2d5aec0	912262732382

Figure 6 Internet Gateway in 'Detached' state

### 4.3 Attach the Internet Gateway to the VPC

Attached using this command:

```
aws ec2 attach-internet-gateway \  
  
--internet-gateway-id igw-0177f6a2d532f0987 \  
  
--vpc-id vpc-0c4561431355c5a6f
```

Screenshot 7: Successful execution (no output)

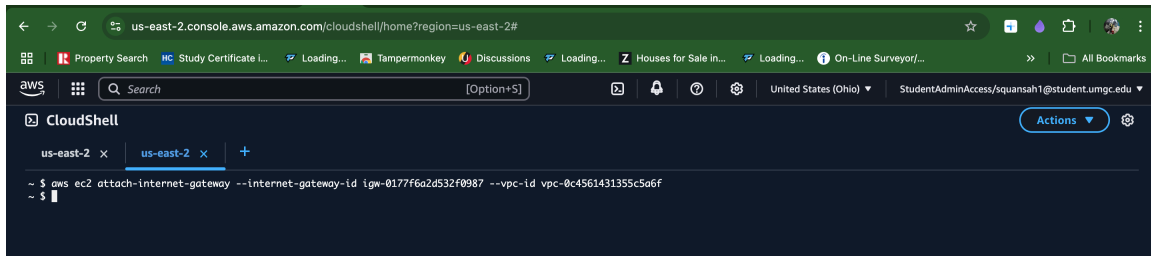


Figure 7 Successful execution (no output)

Verified the gateway attachment:

Screenshot 8: Console showing Internet Gateway in 'Attached' state with correct VPC ID

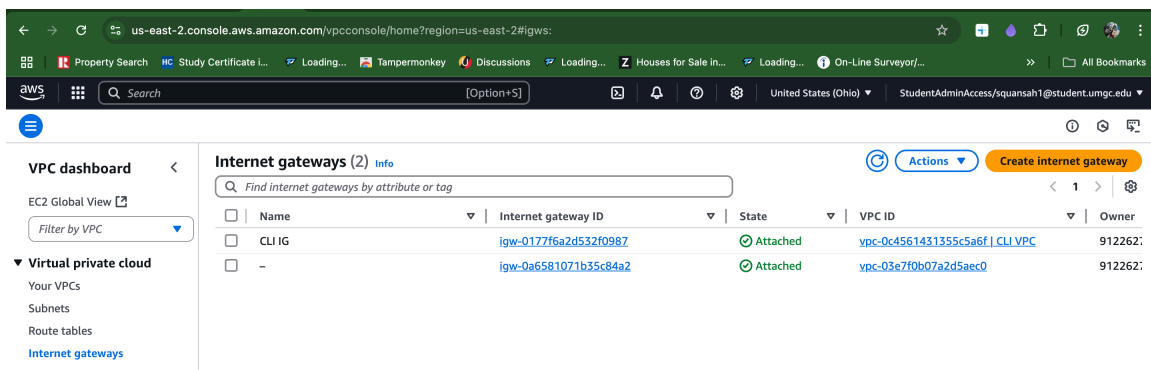


Figure 8 Console showing Internet Gateway in 'Attached' state with correct VPC ID



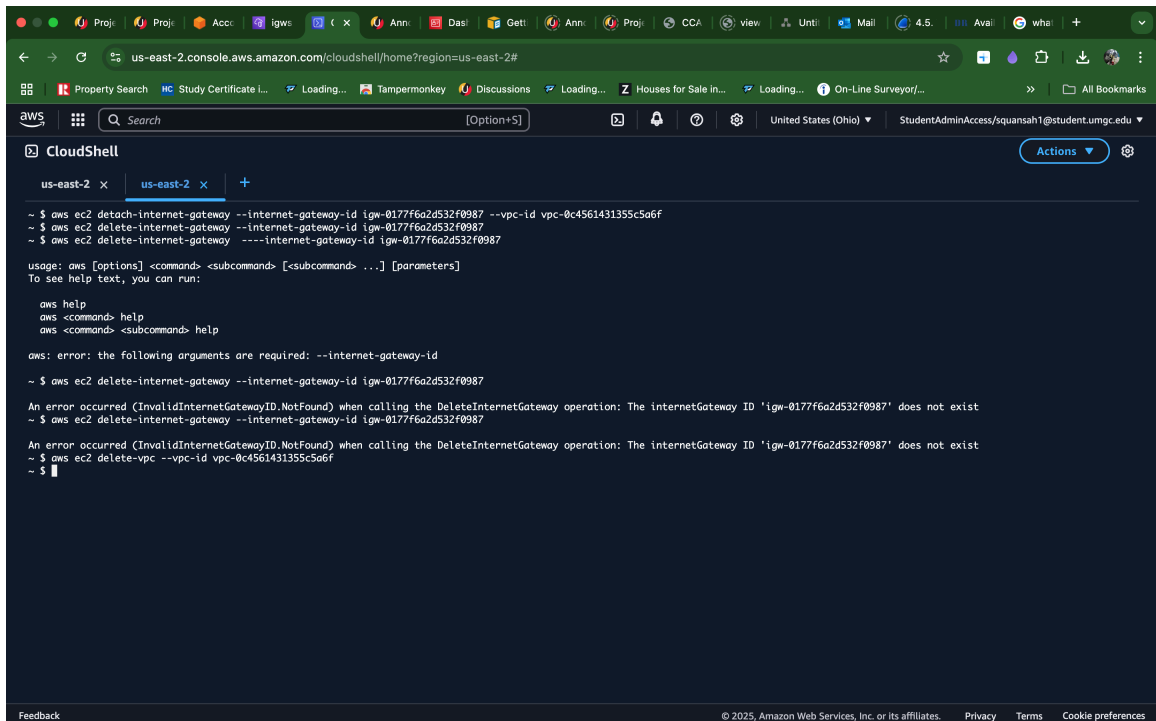
## Cleanup (Recommended)

To avoid incurring charges, I issued the following deletion commands:

```
aws ec2 detach-internet-gateway \  
  
--internet-gateway-id igw-0177f6a2d532f0987 \  
  
--vpc-id vpc-0c4561431355c5a6f
```

```
aws ec2 delete-internet-gateway \  
  
--internet-gateway-id igw-0177f6a2d532f0987
```

```
aws ec2 delete-vpc \  
  
--vpc-id vpc-0c4561431355c5a6f
```



The screenshot shows the AWS CloudShell interface in a web browser. The terminal window displays the following commands and their outputs:

```
~ $ aws ec2 detach-internet-gateway --internet-gateway-id igw-0177f6a2d532f0987 --vpc-id vpc-0c4561431355c5a6f  
~ $ aws ec2 delete-internet-gateway --internet-gateway-id igw-0177f6a2d532f0987  
~ $ aws ec2 delete-internet-gateway --internet-gateway-id igw-0177f6a2d532f0987  
  
usage: aws [options] <command> [<subcommand> ...] [<parameters>]  
To see help text, you can run:  
  
aws help  
aws <command> help  
aws <command> <subcommand> help  
  
aws error: the following arguments are required: --internet-gateway-id  
  
~ $ aws ec2 delete-internet-gateway --internet-gateway-id igw-0177f6a2d532f0987  
An error occurred (InvalidInternetGatewayID.NotFound) when calling the DeleteInternetGateway operation: The internetGateway ID 'igw-0177f6a2d532f0987' does not exist  
~ $ aws ec2 delete-internet-gateway --internet-gateway-id igw-0177f6a2d532f0987  
An error occurred (InvalidInternetGatewayID.NotFound) when calling the DeleteInternetGateway operation: The internetGateway ID 'igw-0177f6a2d532f0987' does not exist  
~ $ aws ec2 delete-vpc --vpc-id vpc-0c4561431355c5a6f  
~ $
```

Figure 9 lab cleanup

## Conclusion

In this lab, I successfully used AWS CloudShell and the AWS CLI to:

- Provision a custom VPC with a defined CIDR block
- Create and tag an Internet Gateway
- Attach the gateway to the VPC

These tasks demonstrate fundamental skills in AWS networking and infrastructure management using command-line tools. This approach is repeatable, scriptable, and essential for DevOps and cloud automation. By understanding how to provision and manage network components from the CLI, I'm better prepared for real-world cloud administration and infrastructure-as-code (IaC) scenarios.