LAB EXAM CC

Name: Maria Abdul Malik
2023-BSE-076
Submitted to: Sir Waqas

Q1 – AWS IAM Setup Using AWS CLI and Console Verification (10 marks)

```
@mariamalik11 /workspaces/lab_exam (main) $ aws iam create-group --group-name Sof
twareEngineering
{
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPAUHXY42RXHCKQUE7IL",
        "Arn": "arn:aws:iam::291506017390:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T08:13:35+00:00"
    }
}
@mariamalik11 /workspaces/lab_exam (main) $
```

q1_group_details.png

```
@mariamalik11 /workspaces/lab_exam (main) $ aws iam get-group --group-name SoftwareEngineering

{
    "Users": [],
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPAUHXY42RXHCKQUE7IL",
        "Arn": "arn:aws:iam::291506017390:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T08:13:35+00:00"
    }
}
@mariamalik11 /workspaces/lab_exam (main) $
```

q1_create_user.png

```
@mariamalik11 /workspaces/lab_exam (main) $ aws iam create-user --user-name maria-abdul-malik

{
    "User": {
        "Path": "/",
        "UserName": "maria-abdul-malik",
        "UserId": "AIDAUHXY42RXAX5HFIMVN",
        "Arn": "arn:aws:iam::291506017390:user/maria-abdul-malik",
        "CreateDate": "2026-01-19T08:15:29+00:00"
    }
}
```

q1_user_details.png

```
@mariamalik11 🖹 /workspaces/lab_exam (main) $ aws iam get-user --user-name maria-abdul-malik

{
    "User": {
        "Path": "/",
        "UserName": "maria-abdul-malik",
        "UserId": "AIDAUHXY42RXAX5HFIMVN",
        "Arn": "arn:aws:iam::291506017390:user/maria-abdul-malik",
        "CreateDate": "2026-01-19T08:15:29+00:00"
    }
}
@mariamalik11 🖹 /workspaces/lab_exam (main) $
```

q1_add_user_to_group.png

```
@mariamalik11 🖹 /workspaces/lab_exam (main) $ aws iam add-user-to-group \
>    --user-name maria-abdul-malik \
>    --group-name SoftwareEngineering
@mariamalik11 🖹 /workspaces/lab_exam (main) $
```

q1_find_admin_policy.png

```
@mariamalik11 🖹 /workspaces/lab_exam (main) $ aws iam list-policies --scope AWS --query "Policies[?PolicyName=='AdministratorAccess']"
[
    {
        "PolicyName": "AdministratorAccess",
        "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
        "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 2,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2015-02-06T18:39:46+00:00",
        "UpdateDate": "2015-02-06T18:39:46+00:00"
    }
]
@mariamalik11 🖹 /workspaces/lab_exam (main) $
```

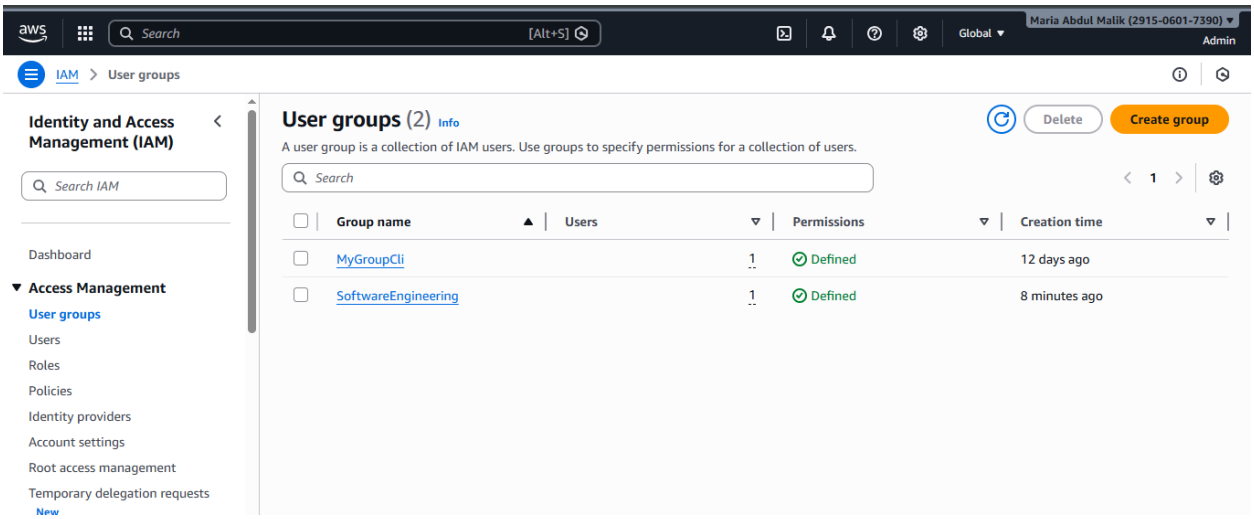
q1_attach_admin_policy.png

```
@mariamalik11 🖹 /workspaces/lab_exam (main) $ aws iam attach-group-policy \
>    --group-name SoftwareEngineering \
>    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
@mariamalik11 🖹 /workspaces/lab_exam (main) $
```

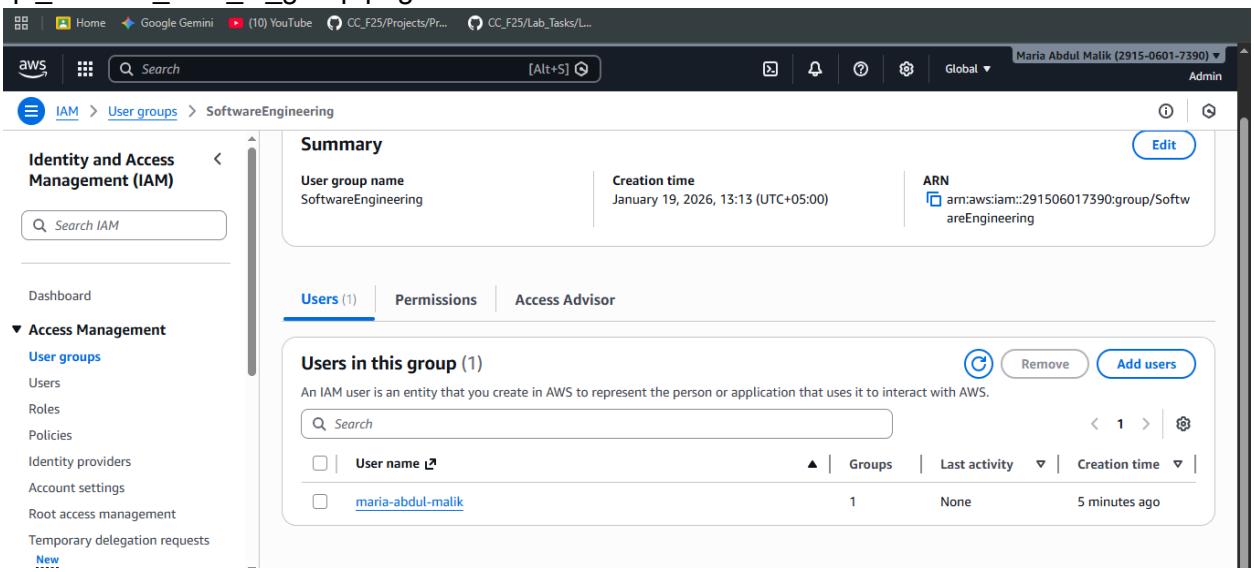
q1_list_group_policies.png

```
@mariamalik11 🖹 /workspaces/lab_exam (main) $ aws iam list-attached-group-policies
 --group-name SoftwareEngineering
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        }
    ]
}
@mariamalik11 🖹 /workspaces/lab_exam (main) $
```

q1_console_group.png



q1_console_user_in_group.png



Q2 – Terraform Lab: Simple AWS Environment with Nginx over HTTPS (30 marks)

q2_provider.png

```
@mariamalik11 /workspaces/lab_exam (main) $ cat main.tf
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
    http = {
      source = "hashicorp/http"
    }
  }
}

provider "aws" {
  region  = "me-central-1"
  profile = "default"
}

@mariamalik11 /workspaces/lab_exam (main) $
```

q2_variables.png

```
@mariamalik11 /workspaces/lab_exam (main) $ @mariamalik
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
    http = {
      source = "hashicorp/http"
    }
  }
}

provider "aws" {
  region  = "me-central-1"
  profile = "default"
}

resource "aws_vpc" "myapp_vpc" {
  cidr_block = var.vpc_cidr_block

  tags = {
    Name = "${var.env_prefix}-vpc"
  }
}

resource "aws_subnet" "myapp_subnet" {
  vpc_id            = aws_vpc.myapp_vpc.id
  cidr_block        = var.subnet_cidr_block
  availability_zone = var.availability_zone

  tags = {
    Name = "${var.env_prefix}-subnet-1"
  }
}

resource "aws_internet_gateway" "myapp_igw" {
  vpc_id = aws_vpc.myapp_vpc.id

  tags = {
    Name = "${var.env_prefix}-igw"
  }
}
```

q2_vpc_subnet.png

```
resource "aws_default_route_table" "myapp_rt" {
  default_route_table_id = aws_vpc.myapp_vpc.default_route_table_id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.myapp_igw.id
  }

  tags = {
    Name = "${var.env_prefix}-rt"
  }
}

data "http" "my_ip" {
  url = "https://icanhazip.com"
}

locals {
  my_ip = "${chomp(data.http.my_ip.response_body)}/32"
}

resource "aws_default_security_group" "default_sg" {
  vpc_id = aws_vpc.myapp_vpc.id

  ingress {
    protocol    = "tcp"
    from_port   = 22
    to_port     = 22
    cidr_blocks = [locals.my_ip]
  }

  ingress {
    protocol    = "tcp"
    from_port   = 80
    to_port     = 80
    cidr_blocks = ["0.0.0.0/0"]
  }

  ingress {
    protocol    = "tcp"
    from_port   = 443
    to_port     = 443
    cidr_blocks = ["0.0.0.0/0"]
  }
```

Q2_igw_route_table.png
Q2_http_and_locals.png
q2_ec2_resource.png

```
    to_port     = 80
    cidr_blocks = ["0.0.0.0/0"]
  }

  ingress {
    protocol    = "tcp"
    from_port   = 443
    to_port     = 443
    cidr_blocks = ["0.0.0.0/0"]
  }

  egress {
    protocol    = "-1"
    from_port   = 0
    to_port     = 0
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    Name = "${var.env_prefix}-default-sg"
  }
}

resource "aws_key_pair" "serverkey" {
  key_name   = "serverkey"
  public_key = file("~/.ssh/id_ed25519.pub")
}

resource "aws_instance" "myapp_ec2" {
  ami                         = "ami-0d7a109bf30624c99" # Amazon Linux 2023 (me-central-1)
  instance_type               = var.instance_type
  subnet_id                   = aws_subnet.myapp_subnet.id
  vpc_security_group_ids       = [aws_default_security_group.default_sg.id]
  availability_zone           = var.availability_zone
  associate_public_ip_address = true
  key_name                    = aws_key_pair.serverkey.key_name
  user_data                   = file("entry-script.sh")

  tags = {
    Name = "${var.env_prefix}-ec2-instance"
  }
}

@mariamalik11 /workspaces/lab_exam (main) $
```

q2_entry_script.png

```
@mariamalik11 ⍰ /workspaces/lab_exam (main) $ @mariamalik11 ⍰ /workspaces/lab_exa
 (main) $ cat entry-script.sh
#!/bin/bash

dnf update -y
dnf install nginx openssl -y

mkdir -p /etc/nginx/ssl

openssl req -x509 -nodes -days 365 \
  -newkey rsa:2048 \
  -keyout /etc/nginx/ssl/nginx.key \
  -out /etc/nginx/ssl/nginx.crt \
  -subj "/CN=localhost"

cat <<EOF > /etc/nginx/conf.d/terraform.conf
server {
    listen 80;
    return 301 https://\$host\$request_uri;
}

server {
    listen 443 ssl;
    ssl_certificate /etc/nginx/ssl/nginx.crt;
    ssl_certificate_key /etc/nginx/ssl/nginx.key;

    location / {
        root /usr/share/nginx/html;
        index index.html;
    }
}
EOF

echo "<h1>This is Maria Abdul Malik's Terraform environment</h1>" > /usr/share/ng
nx/html/index.html

systemctl enable nginx
systemctl restart nginx

@mariamalik11 ⍰ /workspaces/lab_exam (main) $
```

Q2_tfvars_or_vars.png

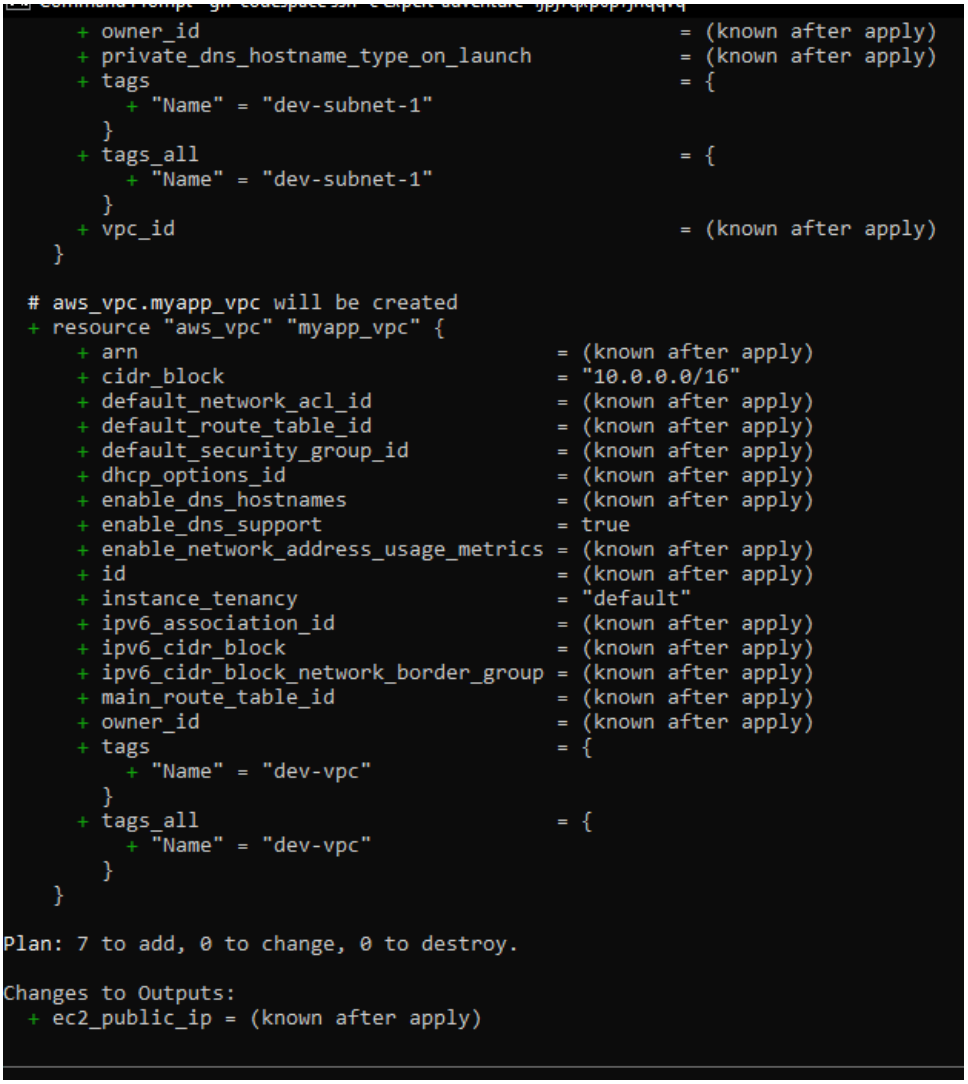
```
@mariamalik11 ⍰ /workspaces/lab_exam (main) $ cat terraform.tfvars
vpc_cidr_block    = "10.0.0.0/16"
subnet_cidr_block = "10.0.10.0/24"
availability_zone = "me-central-1a"
env_prefix        = "dev"
instance_type     = "t3.micro"

@mariamalik11 ⍰ /workspaces/lab_exam (main) $
```

q2_terraform_init.png

```
@mariamalik11 🖹 /workspaces/lab_exam (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Finding latest version of hashicorp/http...
- Installing hashicorp/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (signed by HashiCorp)
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
@mariamalik11 🖹 /workspaces/lab_exam (main) $
```

q2_terraform_plan.png

```
@mariamalik11 🖹 /workspaces/lab_exam (main) $ terraform plan
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_default_route_table.myapp_rt will be created
  + resource "aws_default_route_table" "myapp_rt" {
      + arn                    = (known after apply)
      + default_route_table_id = (known after apply)
      + id                     = (known after apply)
      + owner_id               = (known after apply)
      + route                  = [
          + {
              + cidr_block                = "0.0.0.0/0"
              + gateway_id                = (known after apply)
              # (10 unchanged attributes hidden)
            },
        ]
      + tags                   = {
          + "Name" = "dev-rt"
        }
      + tags_all               = {
          + "Name" = "dev-rt"
        }
      + vpc_id                 = (known after apply)
    }

  # aws_default_security_group.default_sg will be created
  + resource "aws_default_security_group" "default_sg" {
      + arn                    = (known after apply)
      + description            = (known after apply)
      + egress                 = [
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + from_port        = 0
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "-1"
```



```
      + owner_id                                     = (known after apply)
      + private_dns_hostname_type_on_launch          = (known after apply)
      + tags                                         = {
          + "Name" = "dev-subnet-1"
        }
      + tags_all                                     = {
          + "Name" = "dev-subnet-1"
        }
      + vpc_id                                       = (known after apply)
    }

  # aws_vpc.myapp_vpc will be created
  + resource "aws_vpc" "myapp_vpc" {
      + arn                                = (known after apply)
      + cidr_block                         = "10.0.0.0/16"
      + default_network_acl_id             = (known after apply)
      + default_route_table_id             = (known after apply)
      + default_security_group_id          = (known after apply)
      + dhcp_options_id                    = (known after apply)
      + enable_dns_hostnames               = (known after apply)
      + enable_dns_support                 = true
      + enable_network_address_usage_metrics = (known after apply)
      + id                                 = (known after apply)
      + instance_tenancy                   = "default"
      + ipv6_association_id                = (known after apply)
      + ipv6_cidr_block                    = (known after apply)
      + ipv6_cidr_block_network_border_group = (known after apply)
      + main_route_table_id                = (known after apply)
      + owner_id                           = (known after apply)
      + tags                               = {
          + "Name" = "dev-vpc"
        }
      + tags_all                           = {
          + "Name" = "dev-vpc"
        }
    }

Plan: 7 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + ec2_public_ip = (known after apply)
```

Q2_terraform_apply.png

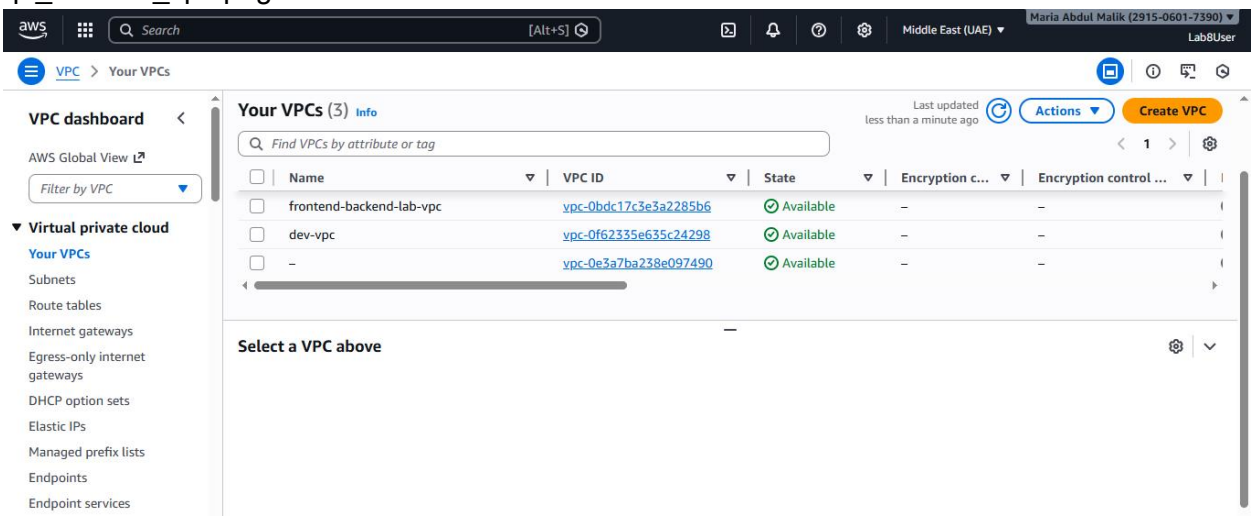
```
@mariamalik11 /workspaces/lab_exam (main) $ terraform apply -auto-approve
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]
data.aws_ami.latest_ubuntu: Reading...
aws_key_pair.serverkey: Refreshing state... [id=serverkey]
aws_vpc.myapp_vpc: Refreshing state... [id=vpc-0f62335e635c24298]
data.aws_ami.latest_ubuntu: Read complete after 1s [id=ami-0745cd130727b73a7]
aws_internet_gateway.myapp_igw: Refreshing state... [id=igw-0d683ea9737790f19]
aws_subnet.myapp_subnet: Refreshing state... [id=subnet-0de6569a2f0e3b744]
aws_default_security_group.default_sg: Refreshing state... [id=sg-0cb81405edf7d7b8
aws_default_route_table.myapp_rt: Refreshing state... [id=rtb-0a0f15e4b7777177c]

Terraform used the selected providers to generate the following execution plan. Re
  + create

Terraform will perform the following actions:

  # aws_instance.myapp_ec2 will be created
  + resource "aws_instance" "myapp_ec2" {
      + ami                                  = "ami-0745cd130727b73a7"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = true
      + availability_zone                    = "me-central-1a"
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
```

```
        + "sg-0cb81405edf7d7b89",
      ]

    + capacity_reservation_specification (known after apply)

    + cpu_options (known after apply)

    + ebs_block_device (known after apply)

    + enclave_options (known after apply)

    + ephemeral_block_device (known after apply)

    + instance_market_options (known after apply)

    + maintenance_options (known after apply)

    + metadata_options (known after apply)

    + network_interface (known after apply)

    + private_dns_name_options (known after apply)

    + root_block_device (known after apply)
  }

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + ec2_public_ip = (known after apply)
aws_instance.myapp_ec2: Creating...
aws_instance.myapp_ec2: Still creating... [00m10s elapsed]
aws_instance.myapp_ec2: Creation complete after 15s [id=i-0819532e60fc

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

ec2_public_ip = "51.112.42.127"
```

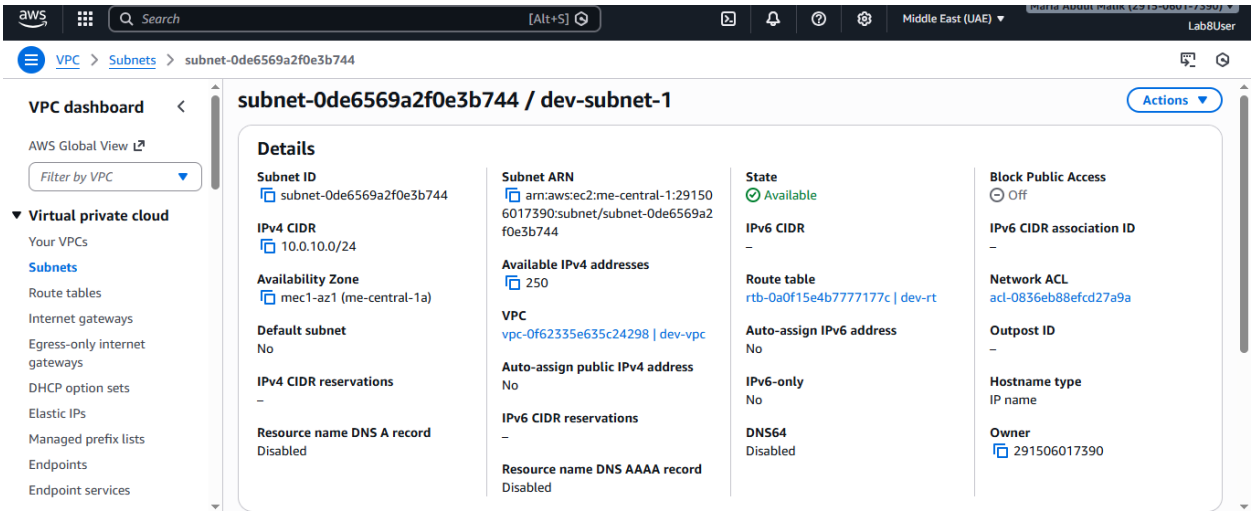
q2_terraform_output.png

```
@mariamalik11 /workspaces/lab_exam (main) $ terraform output
ec2_public_ip = "51.112.42.127"
```

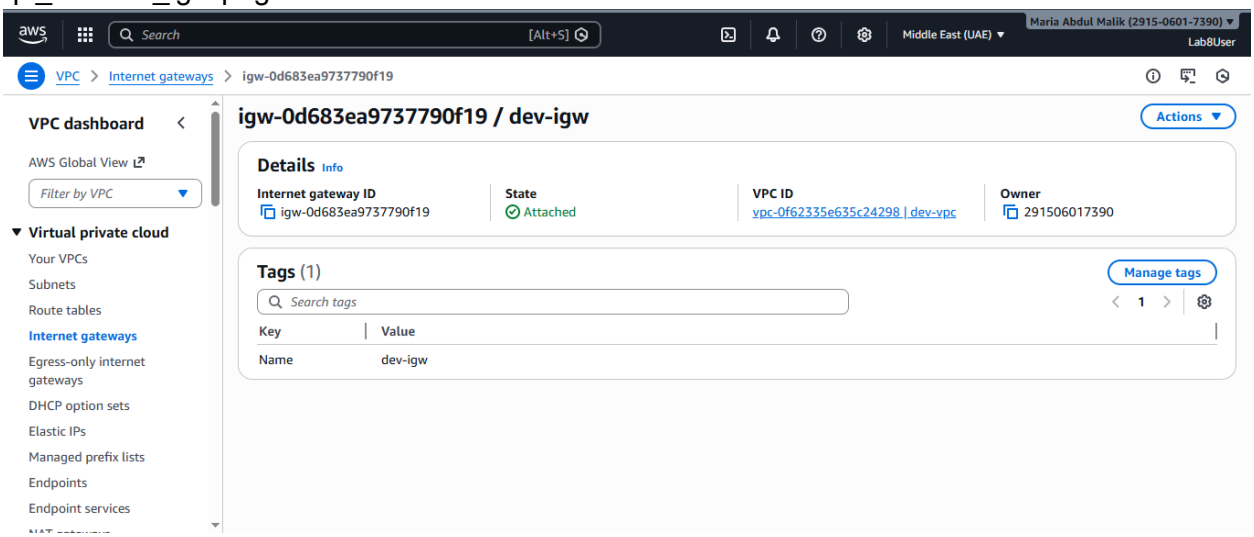
q2_console_vpc.png



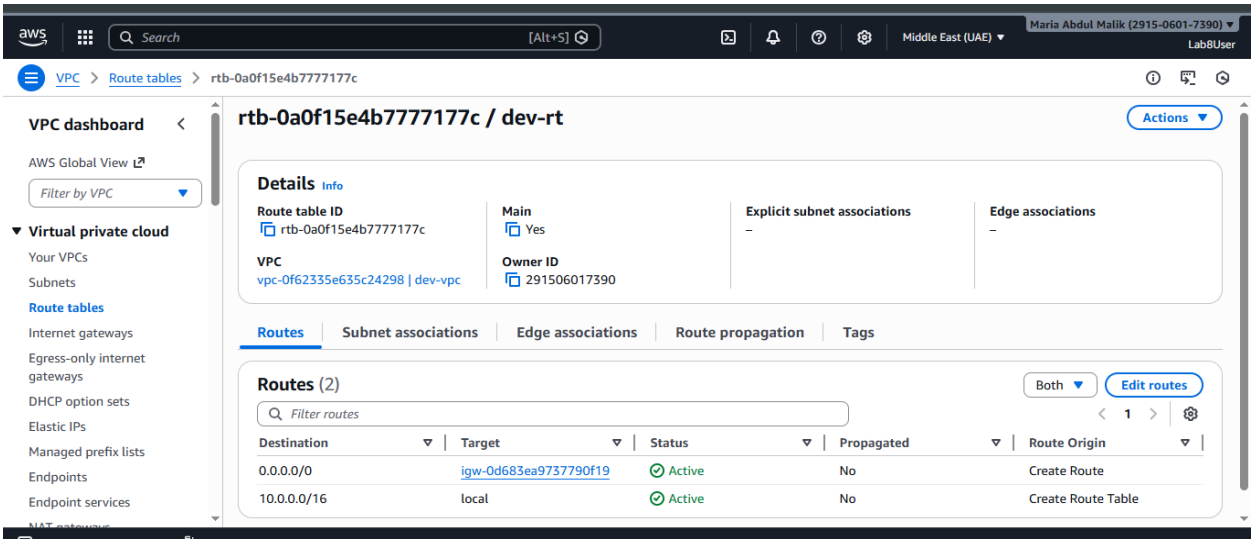
Q2_console_subnet.png



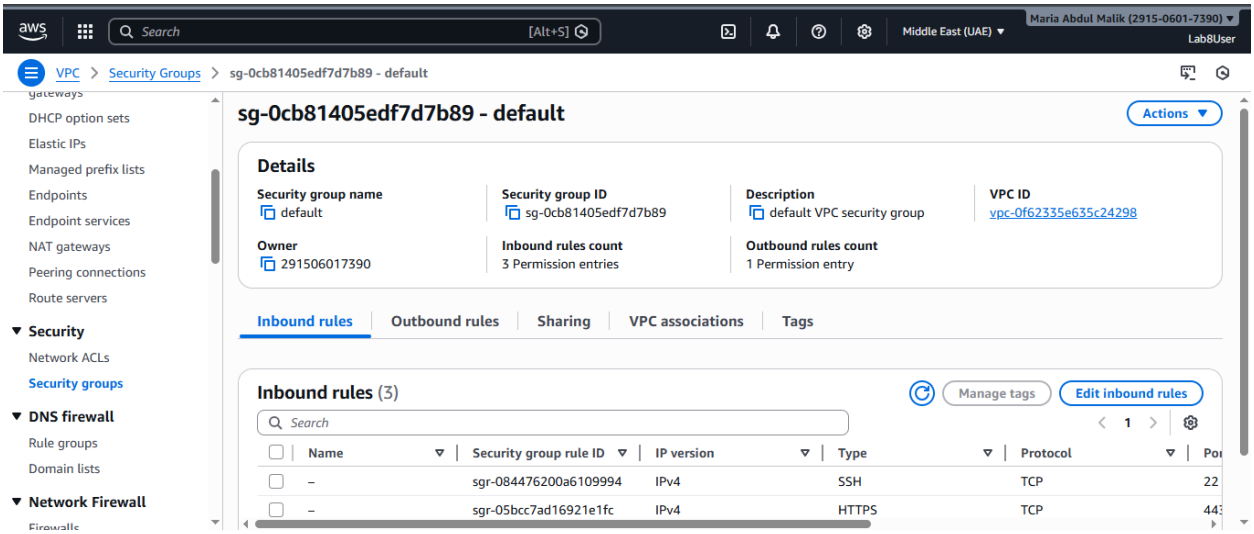
q2_console_igw.png



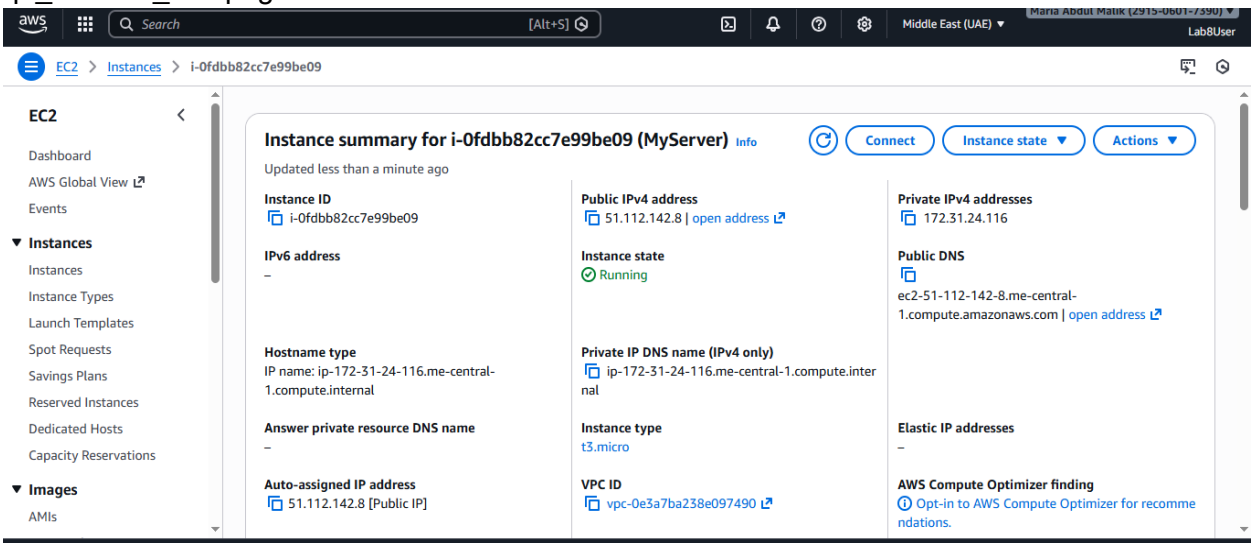
q2_console_route_table.png



q2_console_sg.png



q2_console_ec2.png



q2_https_browser.png