Nayab Khazin

Bse 5 B

046

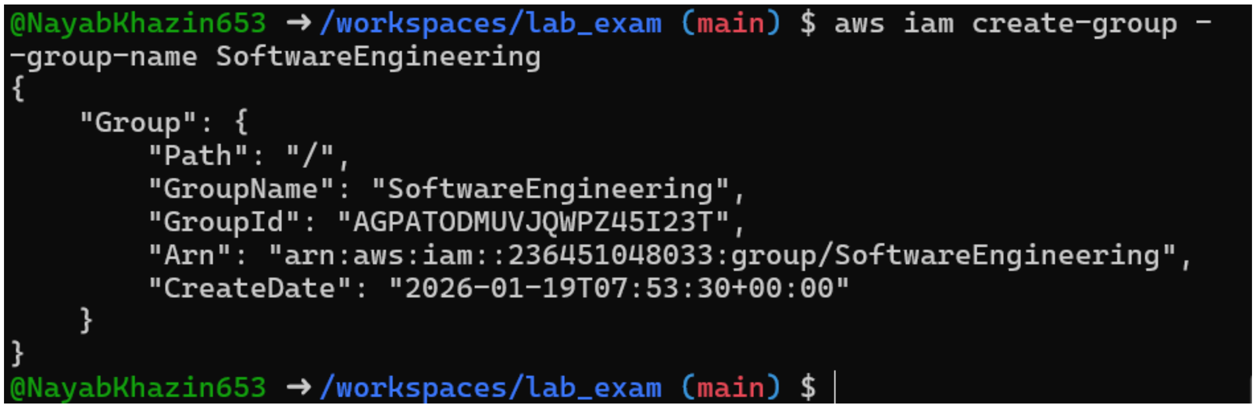
Cloud computing

Submitted to:

Sir waqs saleem

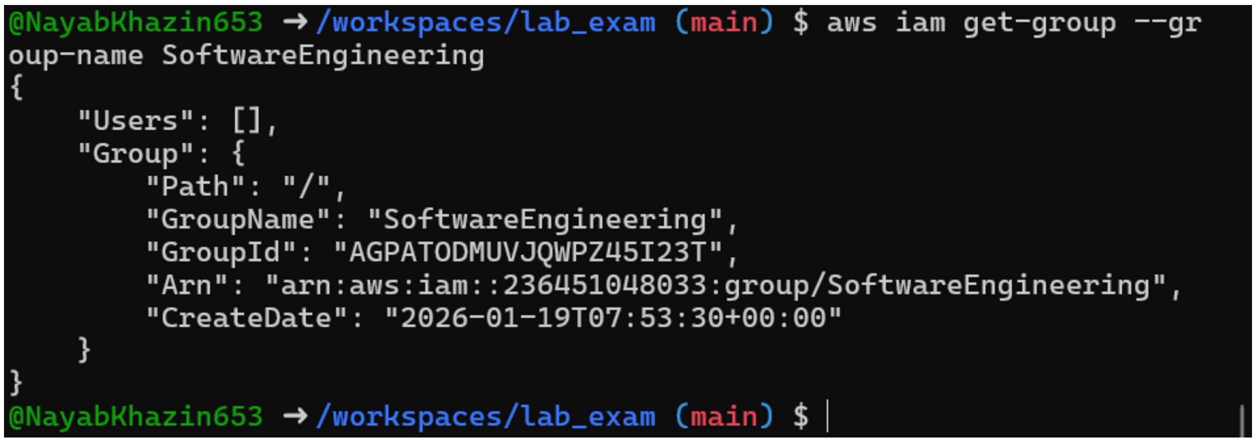
Sir shoaib

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

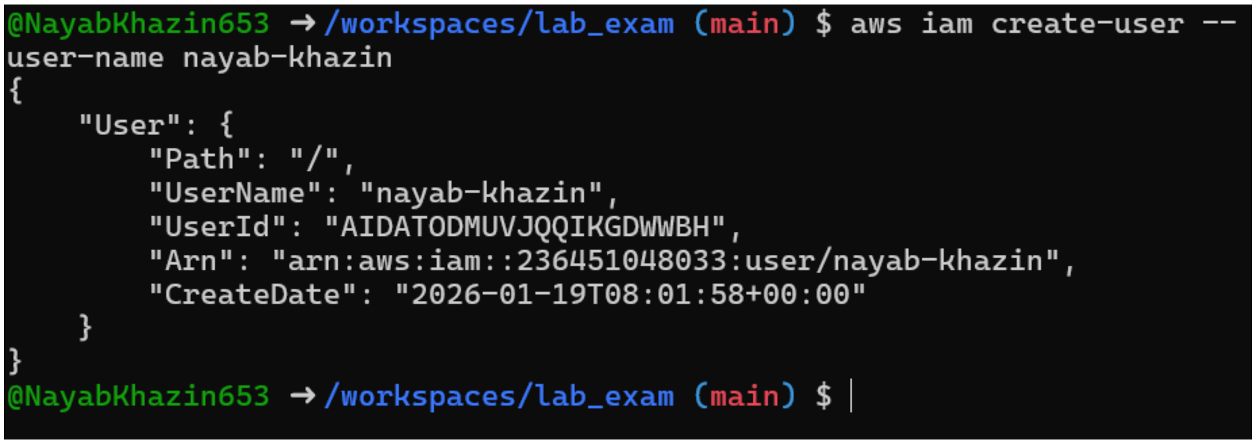


```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam create-group -
-group-name SoftwareEngineering
{
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPATODMUVJQWPZ45I23T",
        "Arn": "arn:aws:iam::236451048033:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:53:30+00:00"
    }
}
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ 
```

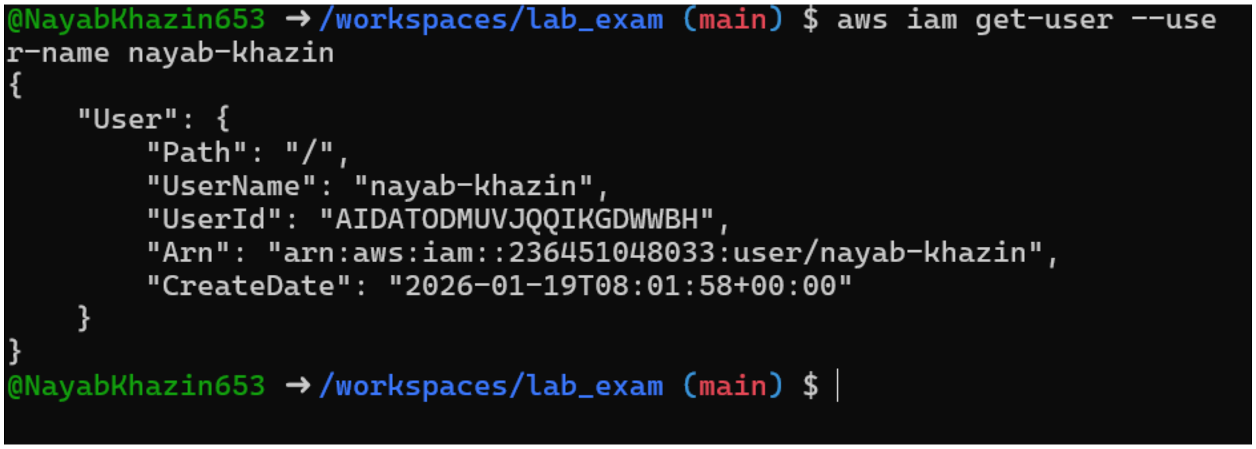
q1_create_group.png


```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam get-group --gr
oup-name SoftwareEngineering
{
    "Users": [],
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPATODMUVJQWPZ45I23T",
        "Arn": "arn:aws:iam::236451048033:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:53:30+00:00"
    }
}
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ 
```

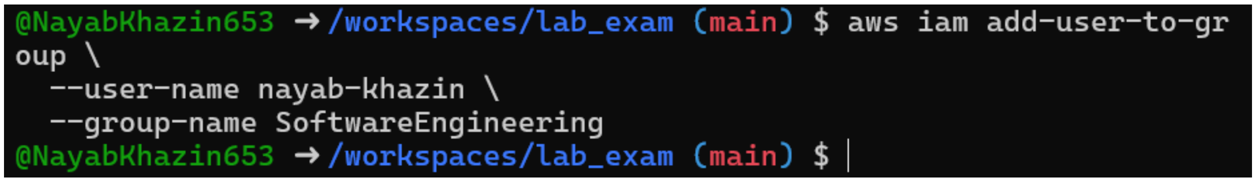
q1_group_details.png


```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam create-user --
user-name nayab-khazin
{
    "User": {
        "Path": "/",
        "UserName": "nayab-khazin",
        "UserId": "AIDATODMUVJQQIKGDWWBH",
        "Arn": "arn:aws:iam::236451048033:user/nayab-khazin",
        "CreateDate": "2026-01-19T08:01:58+00:00"
    }
}
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ 
```

q1_create_user.png

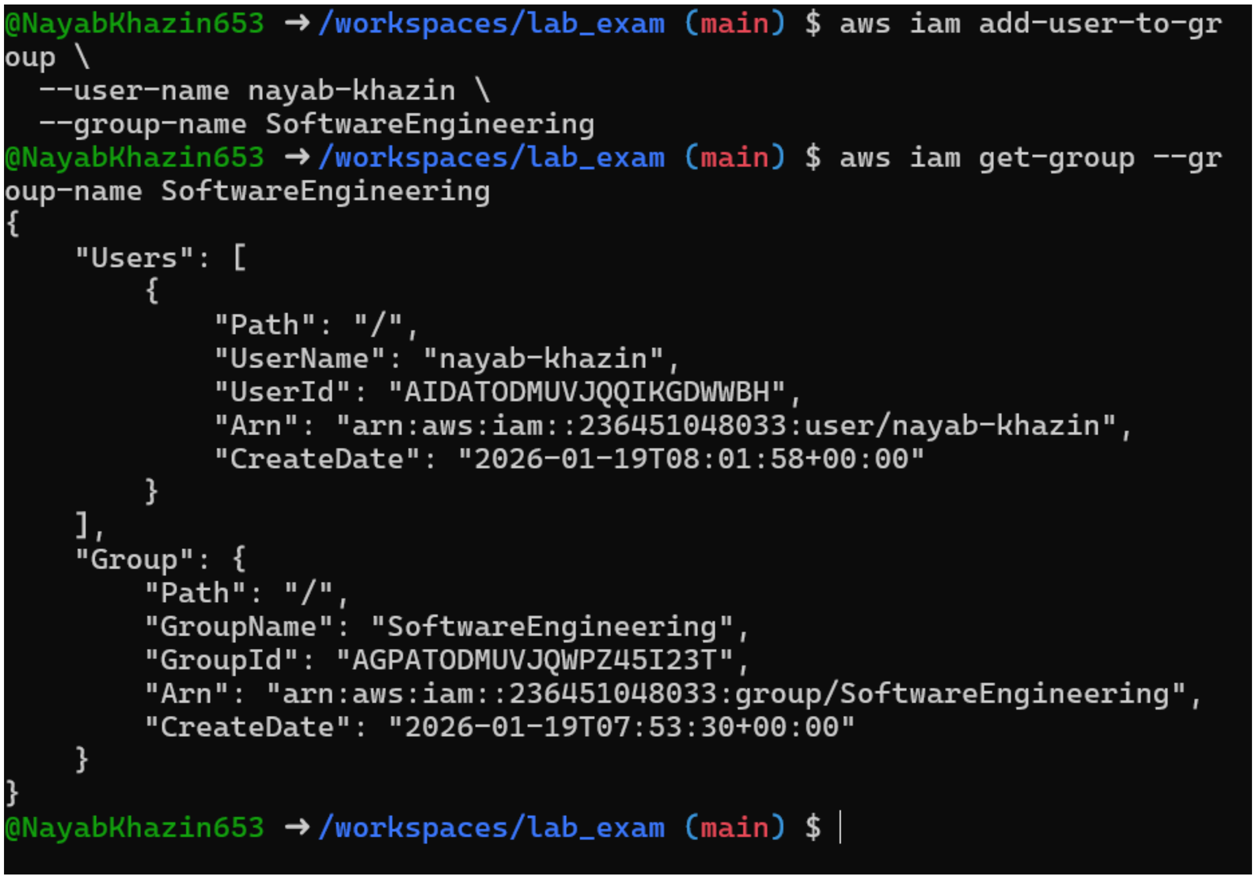

```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam get-user --use
r-name nayab-khazin
{
    "User": {
        "Path": "/",
        "UserName": "nayab-khazin",
        "UserId": "AIDATODMUVJQQIKGDWWBH",
        "Arn": "arn:aws:iam::236451048033:user/nayab-khazin",
        "CreateDate": "2026-01-19T08:01:58+00:00"
    }
}
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ 
```

q1_user_details.png



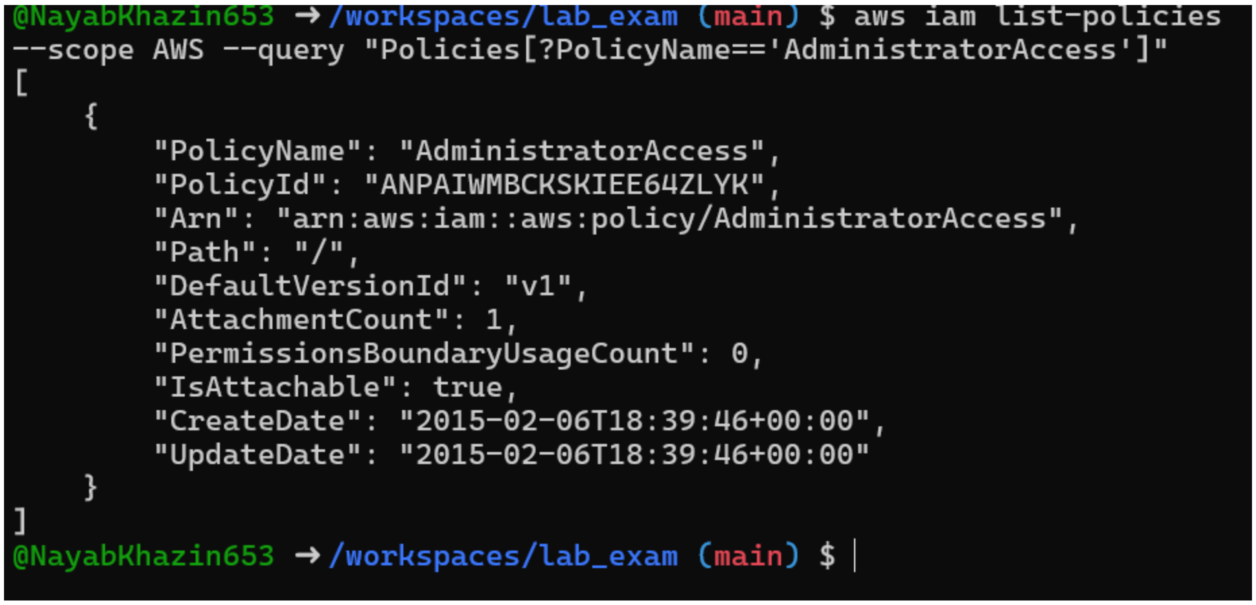
```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam add-user-to-gr
oup \
  --user-name nayab-khazin \
  --group-name SoftwareEngineering
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ 
```

q1_add_user_to_group.png



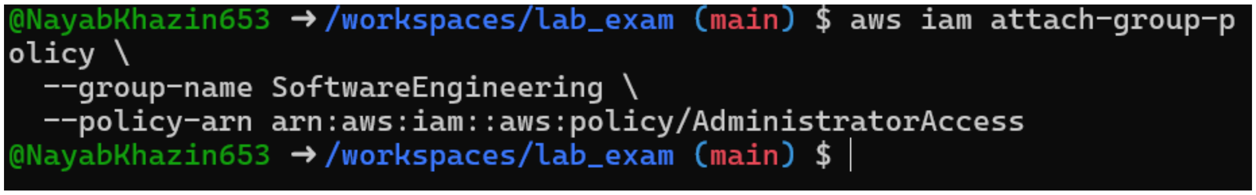
```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam add-user-to-gr
oup \
  --user-name nayab-khazin \
  --group-name SoftwareEngineering
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam get-group --gr
oup-name SoftwareEngineering
{
    "Users": [
        {
            "Path": "/",
            "UserName": "nayab-khazin",
            "UserId": "AIDATODMUVJQQIKGDWWBH",
            "Arn": "arn:aws:iam::236451048033:user/nayab-khazin",
            "CreateDate": "2026-01-19T08:01:58+00:00"
        }
    ],
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPATODMUVJQWPZ45I23T",
        "Arn": "arn:aws:iam::236451048033:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:53:30+00:00"
    }
}
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ 
```

q1_group_membership.png



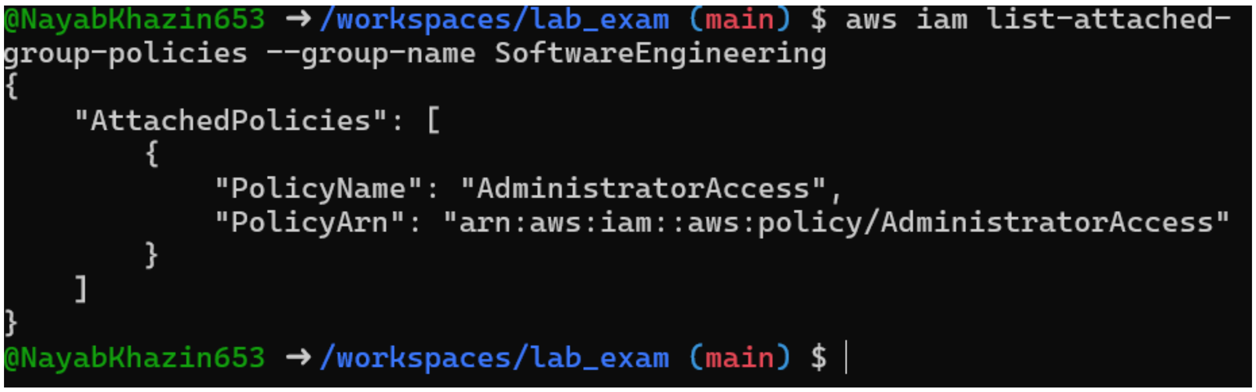
```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam list-policies
--scope AWS --query "Policies[?PolicyName=='AdministratorAccess']"
[
    {
        "PolicyName": "AdministratorAccess",
        "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
        "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 1,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2015-02-06T18:39:46+00:00",
        "UpdateDate": "2015-02-06T18:39:46+00:00"
    }
]
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ |
```

q1_find_admin_policy.png



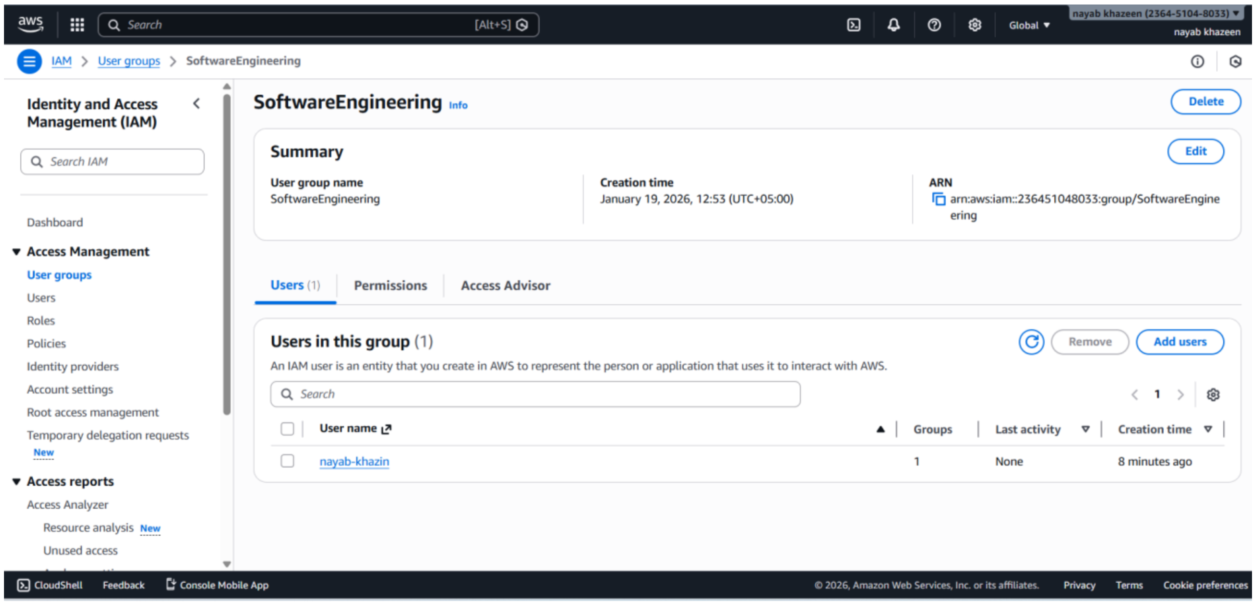
```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam attach-group-p
olicy \
  --group-name SoftwareEngineering \
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ |
```

q1_attach_admin_policy.png



```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ aws iam list-attached-
group-policies --group-name SoftwareEngineering
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        }
    ]
}
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ |
```

q1_list_group_policies.png



SoftwareEngineering Info

Summary

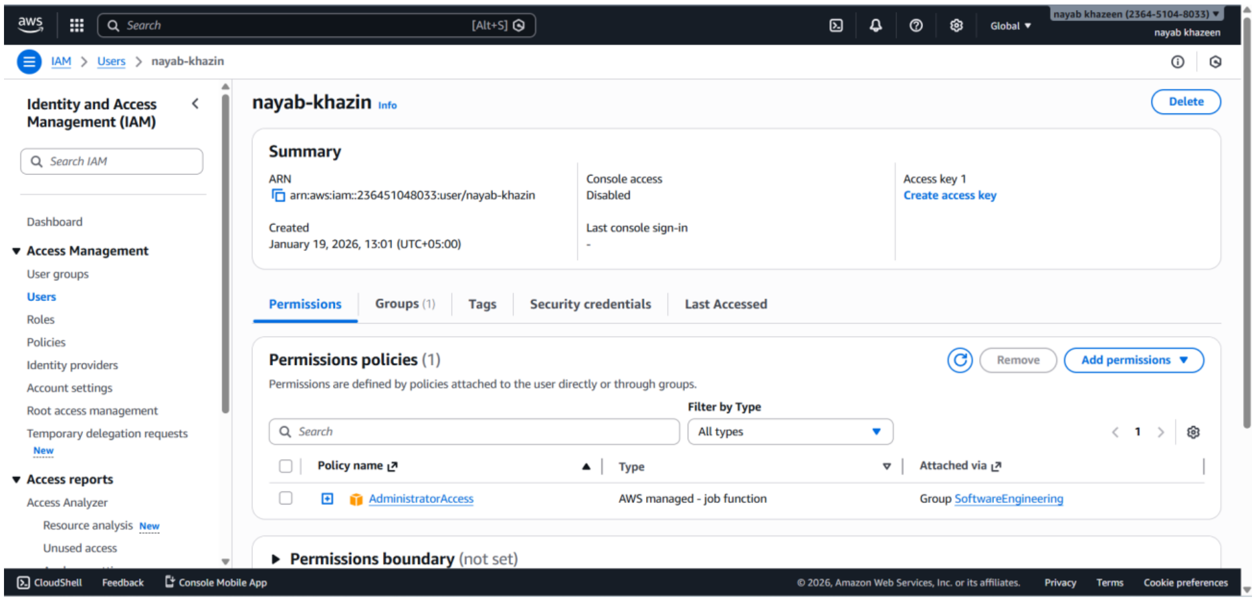
User group name: SoftwareEngineering
Creation time: January 19, 2026, 12:53 (UTC+05:00)
ARN: arn:aws:iam::236451048033:group/SoftwareEngineering

Users in this group (1)

An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS.

User name	Groups	Last activity	Creation time
nayab-khazin	1	None	8 minutes ago

q1_console_group.png



nayab-khazin Info

Summary

ARN: arn:aws:iam::236451048033:user/nayab-khazin
Created: January 19, 2026, 13:01 (UTC+05:00)
Console access: Disabled
Last console sign-in: -
Access key 1: Create access key

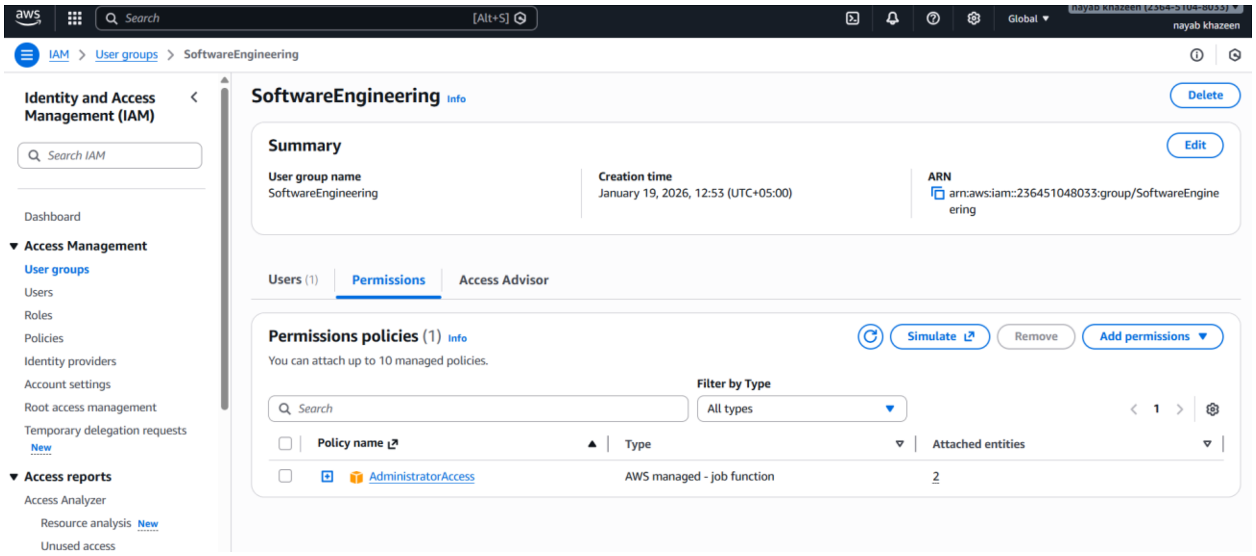
Permissions policies (1)

Permissions are defined by policies attached to the user directly or through groups.

Policy name	Type	Attached via
AdministratorAccess	AWS managed - job function	Group SoftwareEngineering

Permissions boundary (not set)

q1_console_user_in_group.png



SoftwareEngineering

Summary

User group name: SoftwareEngineering

Creation time: January 19, 2026, 12:53 (UTC+05:00)

ARN: arn:aws:iam::236451048033:group/SoftwareEngineering

Permissions policies (1)

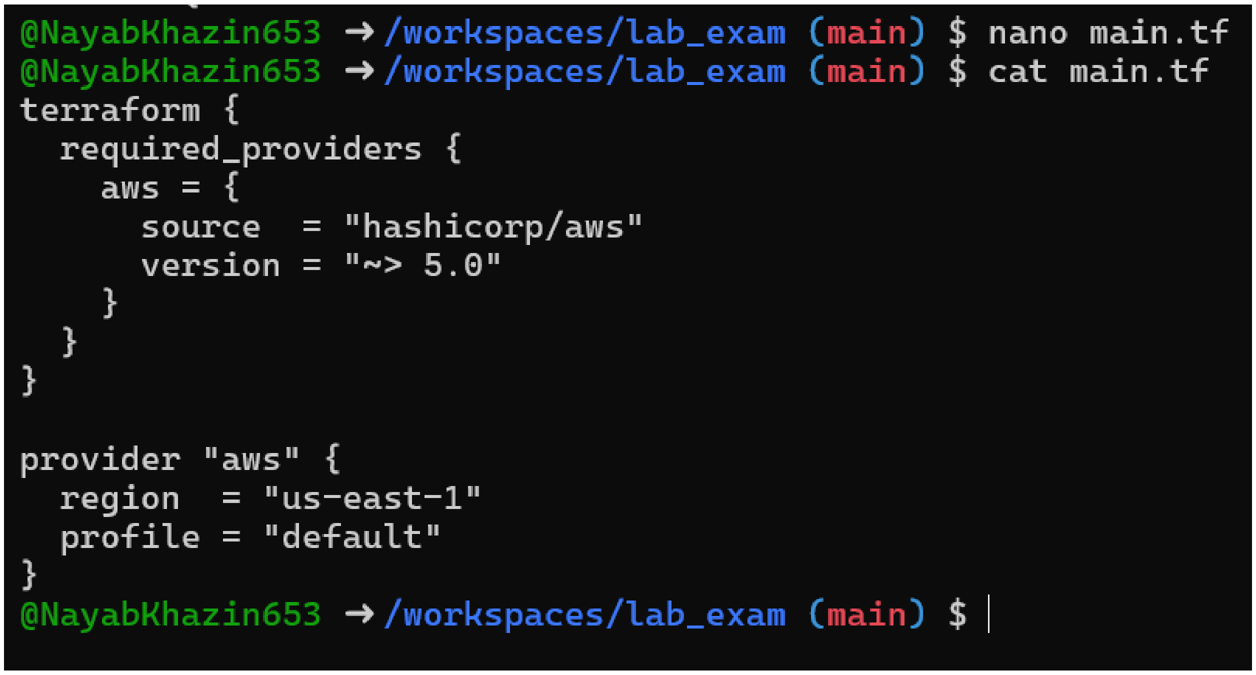
You can attach up to 10 managed policies.

Filter by Type: All types

Policy name	Type	Attached entities
AdministratorAccess	AWS managed - job function	2

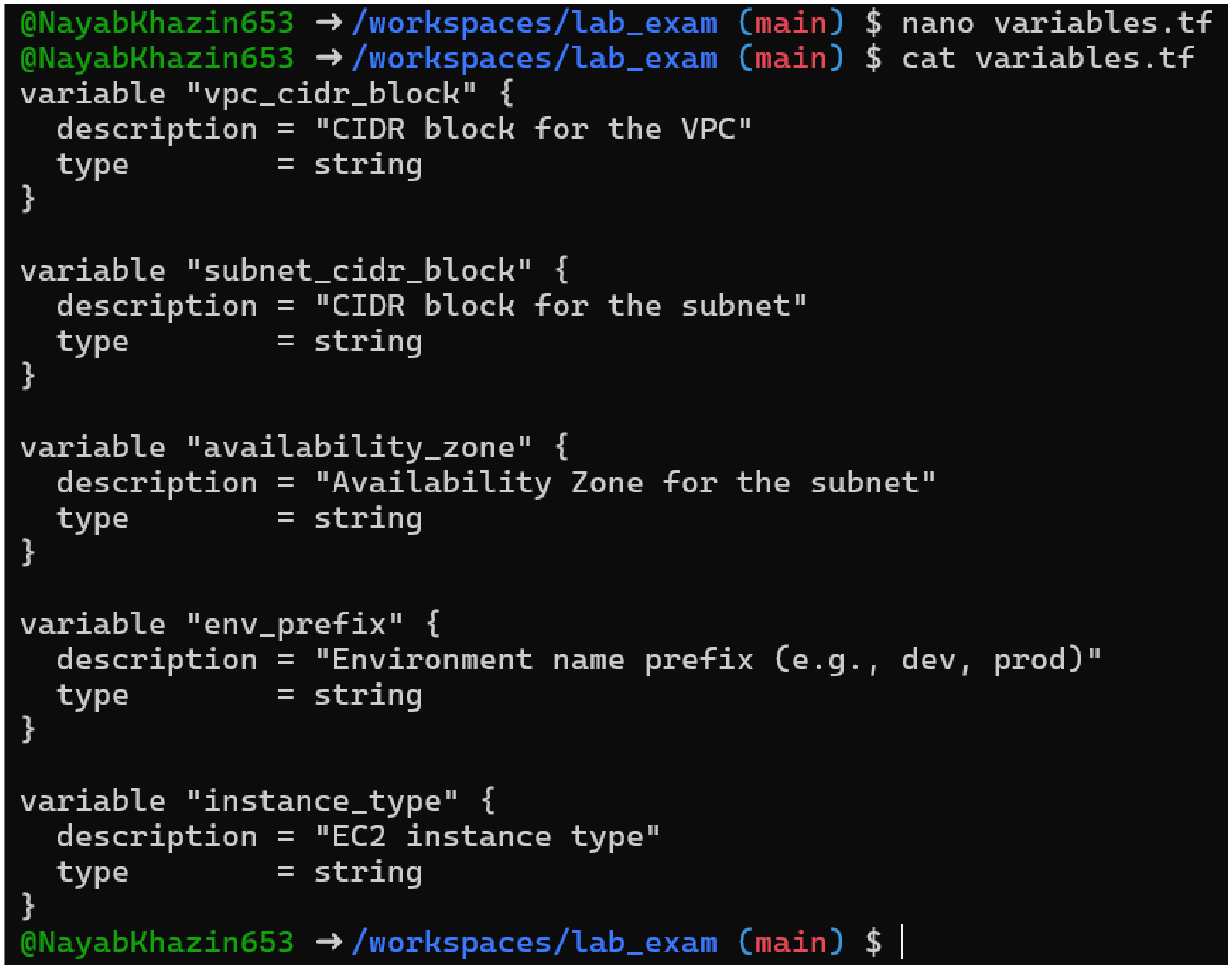
q1_console_group_policy.png

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



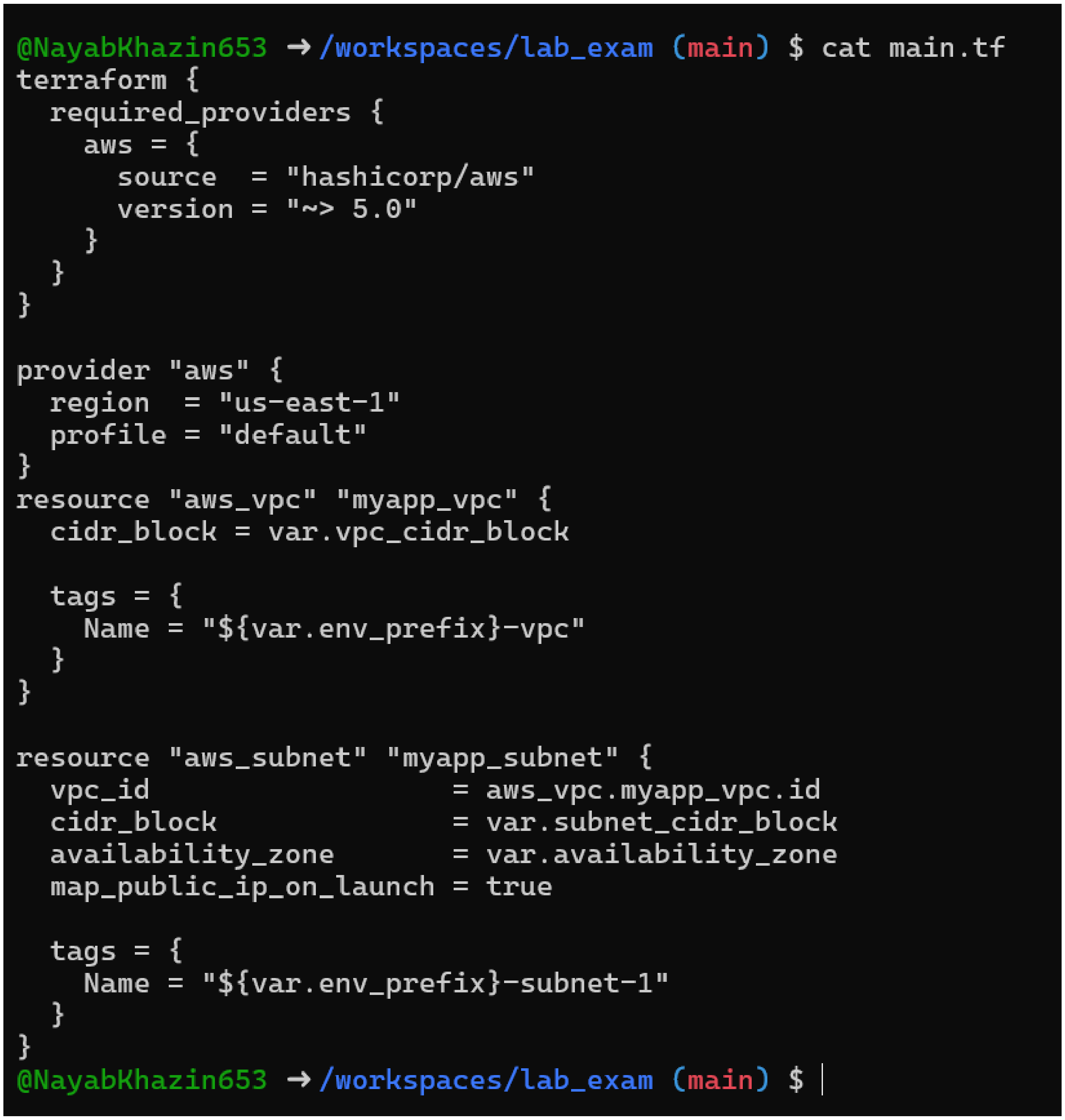
```
@NayabKhazin653 → /workspaces/lab_exam (main) $ nano main.tf
@NayabKhazin653 → /workspaces/lab_exam (main) $ cat main.tf
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

provider "aws" {
  region  = "us-east-1"
  profile = "default"
}
@NayabKhazin653 → /workspaces/lab_exam (main) $
```

q2_provider.png



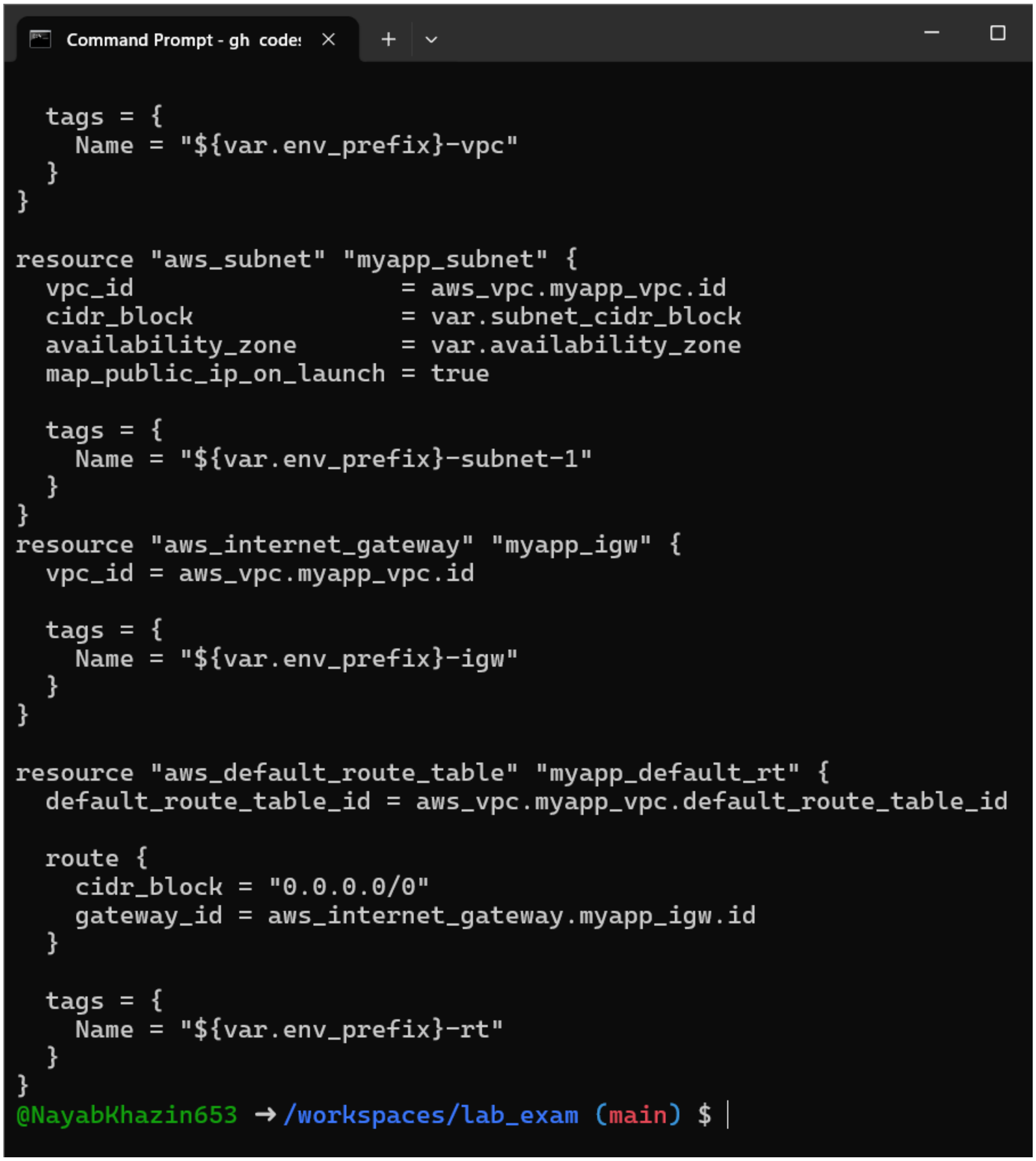
```
@NayabKhazin653 → /workspaces/lab_exam (main) $ nano variables.tf
@NayabKhazin653 → /workspaces/lab_exam (main) $ cat variables.tf
variable "vpc_cidr_block" {
  description = "CIDR block for the VPC"
  type        = string
}

variable "subnet_cidr_block" {
  description = "CIDR block for the subnet"
  type        = string
}

variable "availability_zone" {
  description = "Availability Zone for the subnet"
  type        = string
}

variable "env_prefix" {
  description = "Environment name prefix (e.g., dev, prod)"
  type        = string
}

variable "instance_type" {
  description = "EC2 instance type"
  type        = string
}
@NayabKhazin653 → /workspaces/lab_exam (main) $
```

q2_variables.png



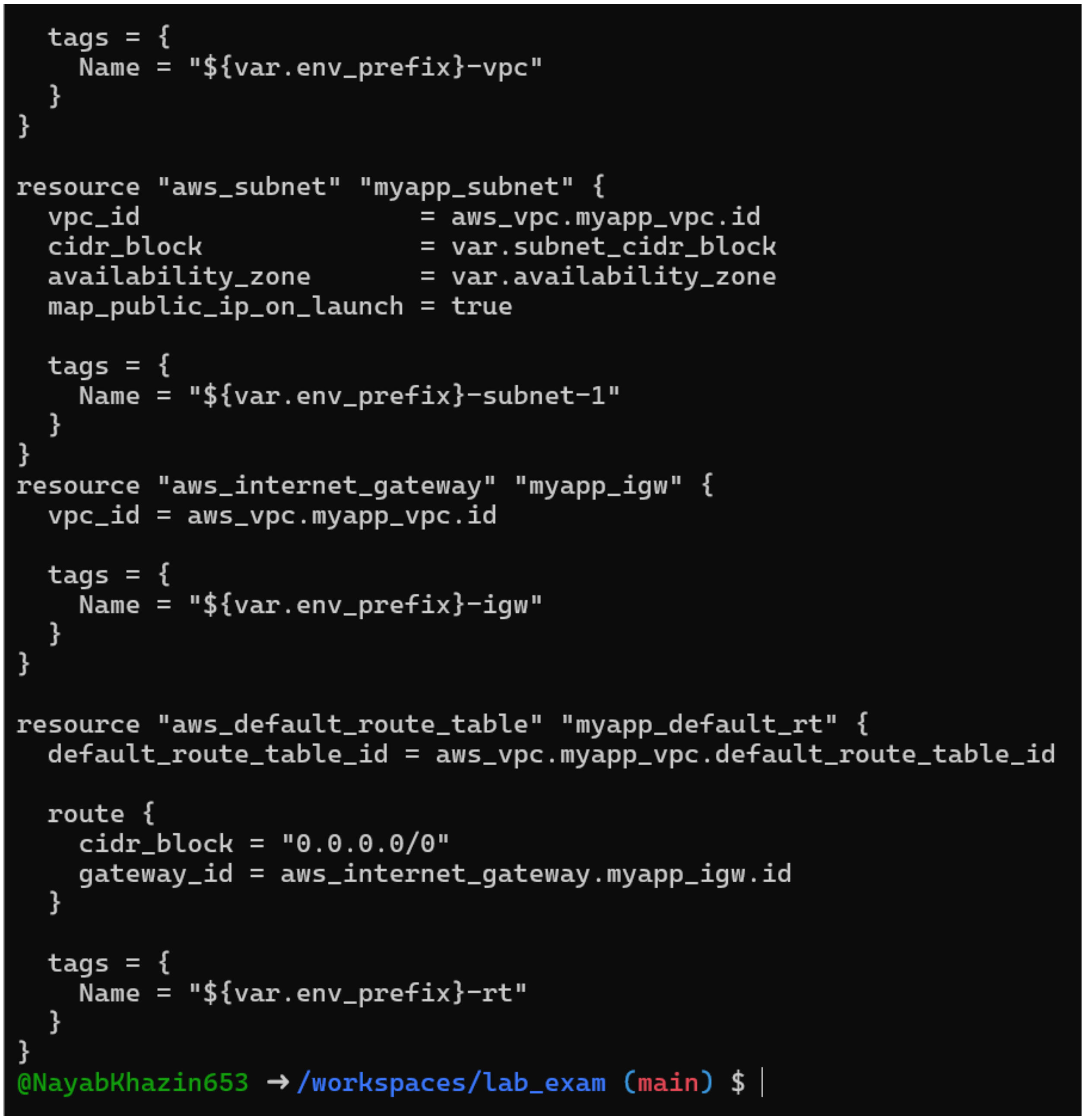
```
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ cat main.tf
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

provider "aws" {
  region  = "us-east-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
  map_public_ip_on_launch = true

  tags = {
    Name = "${var.env_prefix}-subnet-1"
  }
}
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $
```

q2_vpc_subnet.png



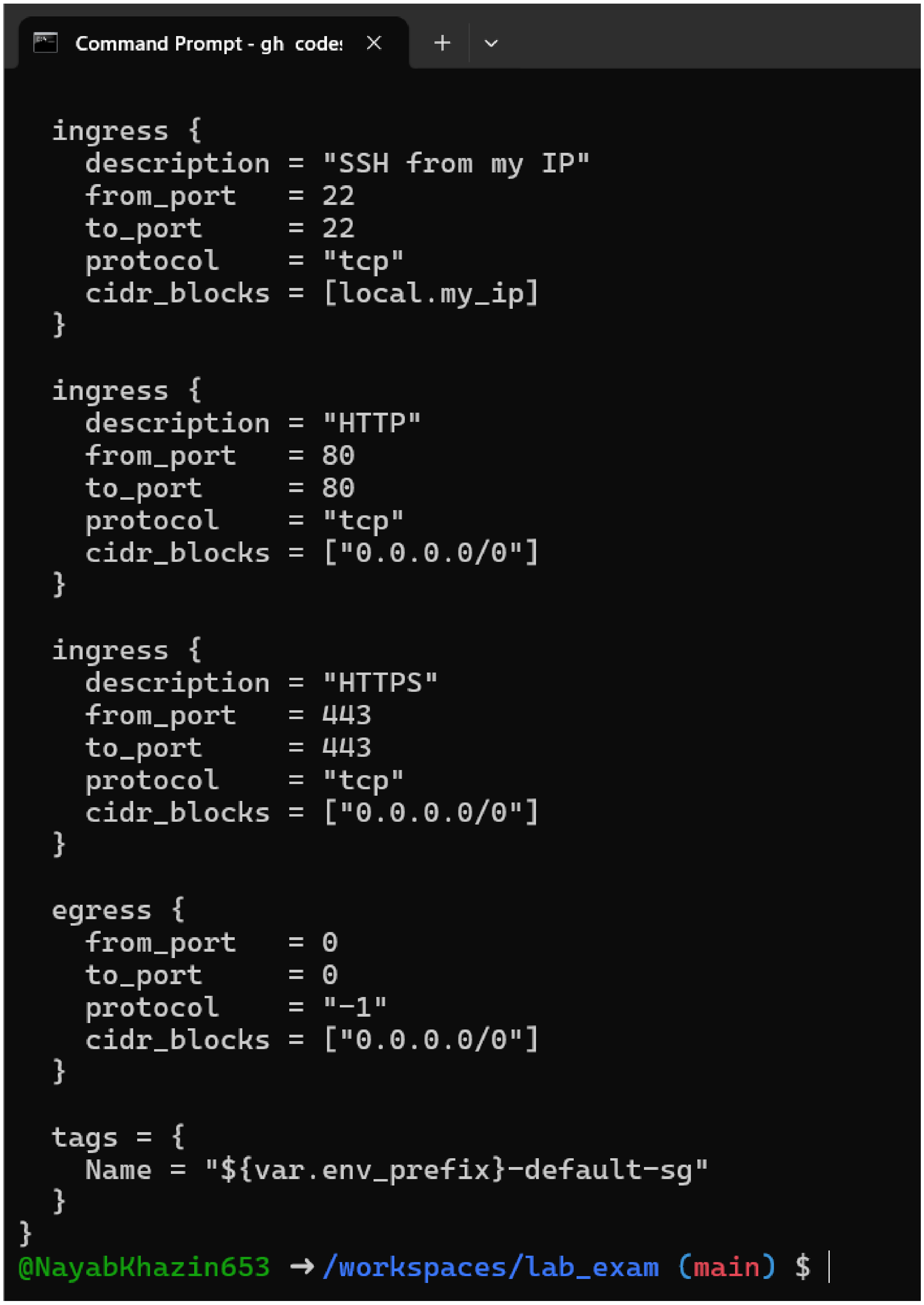
```
  tags = {
    Name = "${var.env_prefix}-vpc"
  }
}

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

  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"
  }
}

resource "aws_default_route_table" "myapp_default_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"
  }
}
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ 
```

q2_igw_route_table.png



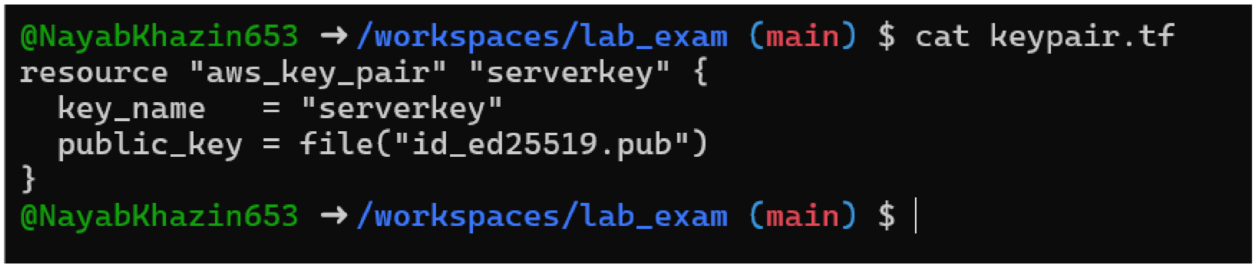
```
  tags = {
    Name = "${var.env_prefix}-vpc"
  }
}

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

  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"
  }
}

resource "aws_default_route_table" "myapp_default_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"
  }
}
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ |
```

q2_http_and_locals.png


Command Prompt - gh code:

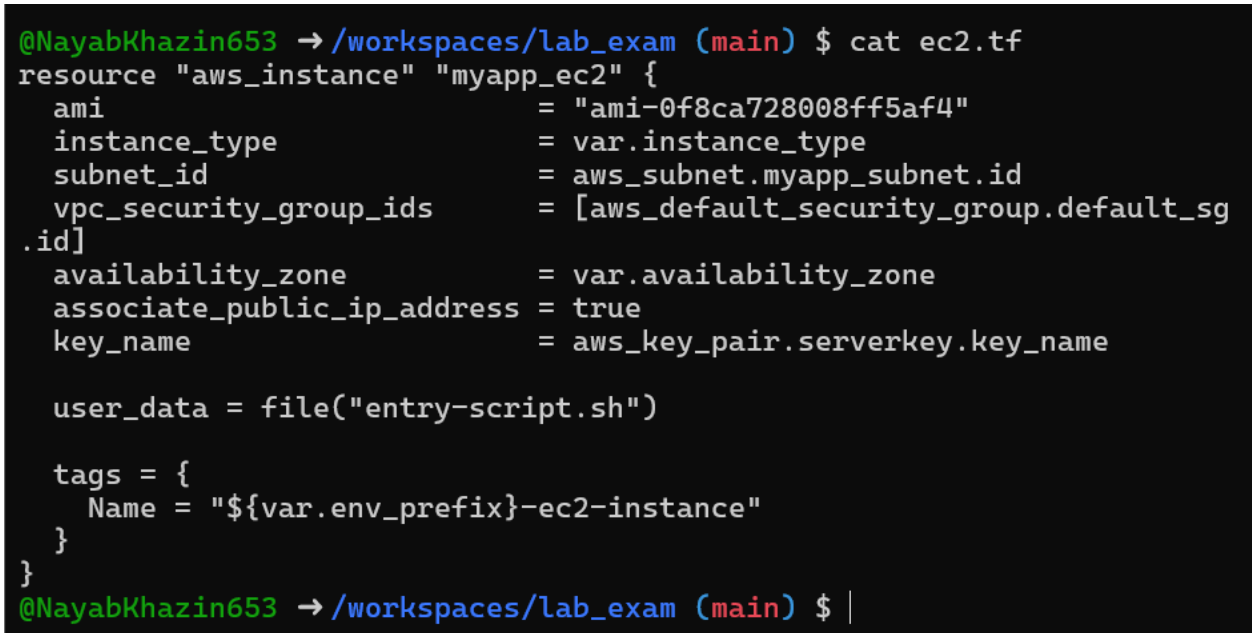
```
  ingress {
    description = "SSH from my IP"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [local.my_ip]
  }

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

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

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

  tags = {
    Name = "${var.env_prefix}-default-sg"
  }
}
@NayabKhazin653 → /workspaces/lab_exam (main) $
```

q2_default_sg.png



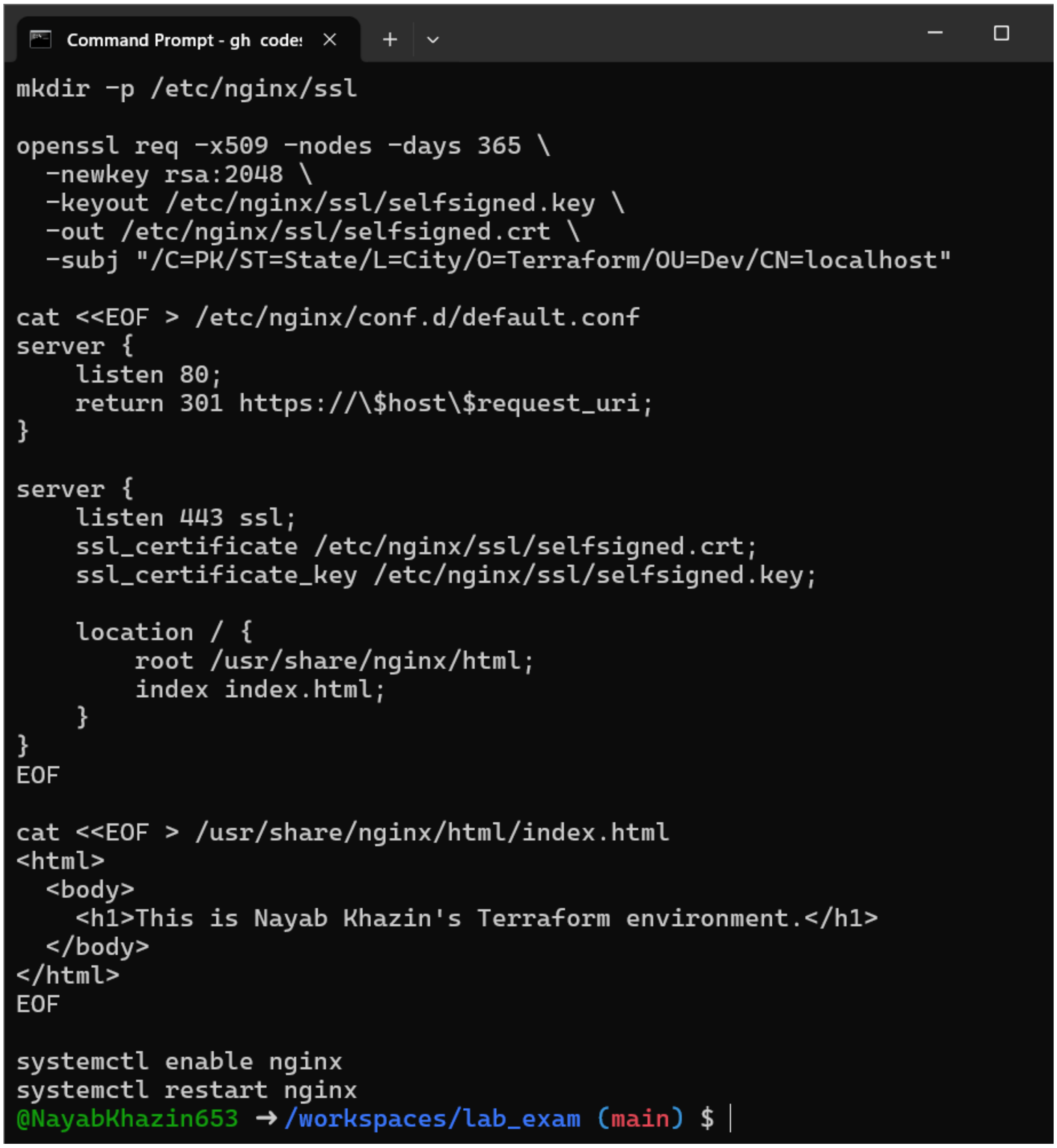
```
@NayabKhazin653 → /workspaces/lab_exam (main) $ cat keypair.tf
resource "aws_key_pair" "serverkey" {
  key_name   = "serverkey"
  public_key = file("id_ed25519.pub")
}
@NayabKhazin653 → /workspaces/lab_exam (main) $ 
```

q2_keypair.png



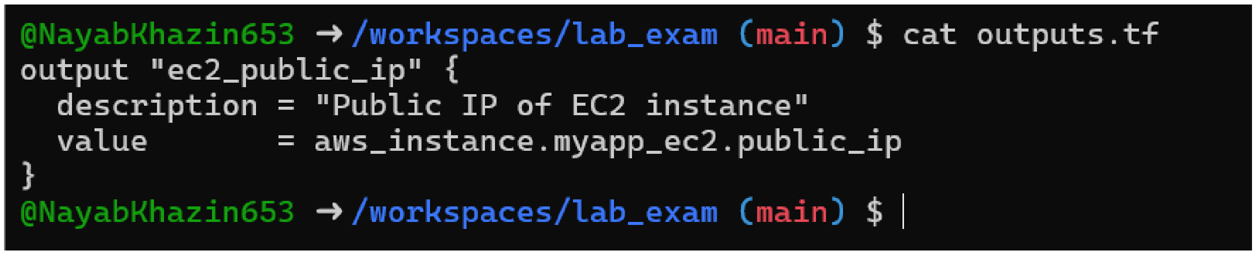
```
@NayabKhazin653 → /workspaces/lab_exam (main) $ cat ec2.tf
resource "aws_instance" "myapp_ec2" {
  ami                         = "ami-0f8ca728008ff5af4"
  instance_type               = var.instance_type
  subnet_id                   = aws_subnet.myapp_subnet.id
  vpc_security_group_ids      = [aws_default_security_group.default_sg
.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"
  }
}
@NayabKhazin653 → /workspaces/lab_exam (main) $ 
```

q2_ec2_resource.png



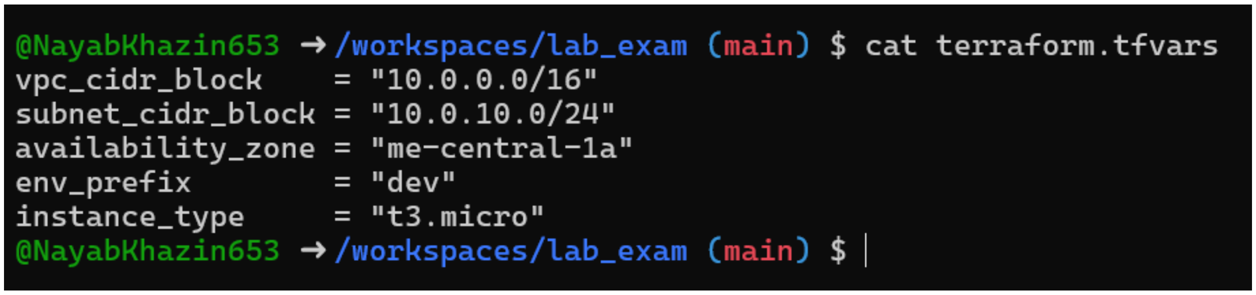
```
mkdir -p /etc/nginx/ssl

openssl req -x509 -nodes -days 365 \
  -newkey rsa:2048 \
  -keyout /etc/nginx/ssl/selfsigned.key \
  -out /etc/nginx/ssl/selfsigned.crt \
  -subj "/C=PK/ST=State/L=City/O=Terraform/OU=Dev/CN=localhost"

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

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

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

cat <<EOF > /usr/share/nginx/html/index.html
<html>
  <body>
    <h1>This is Nayab Khazin's Terraform environment.</h1>
  </body>
</html>
EOF

systemctl enable nginx
systemctl restart nginx
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ 
```

q2_entry_script.png



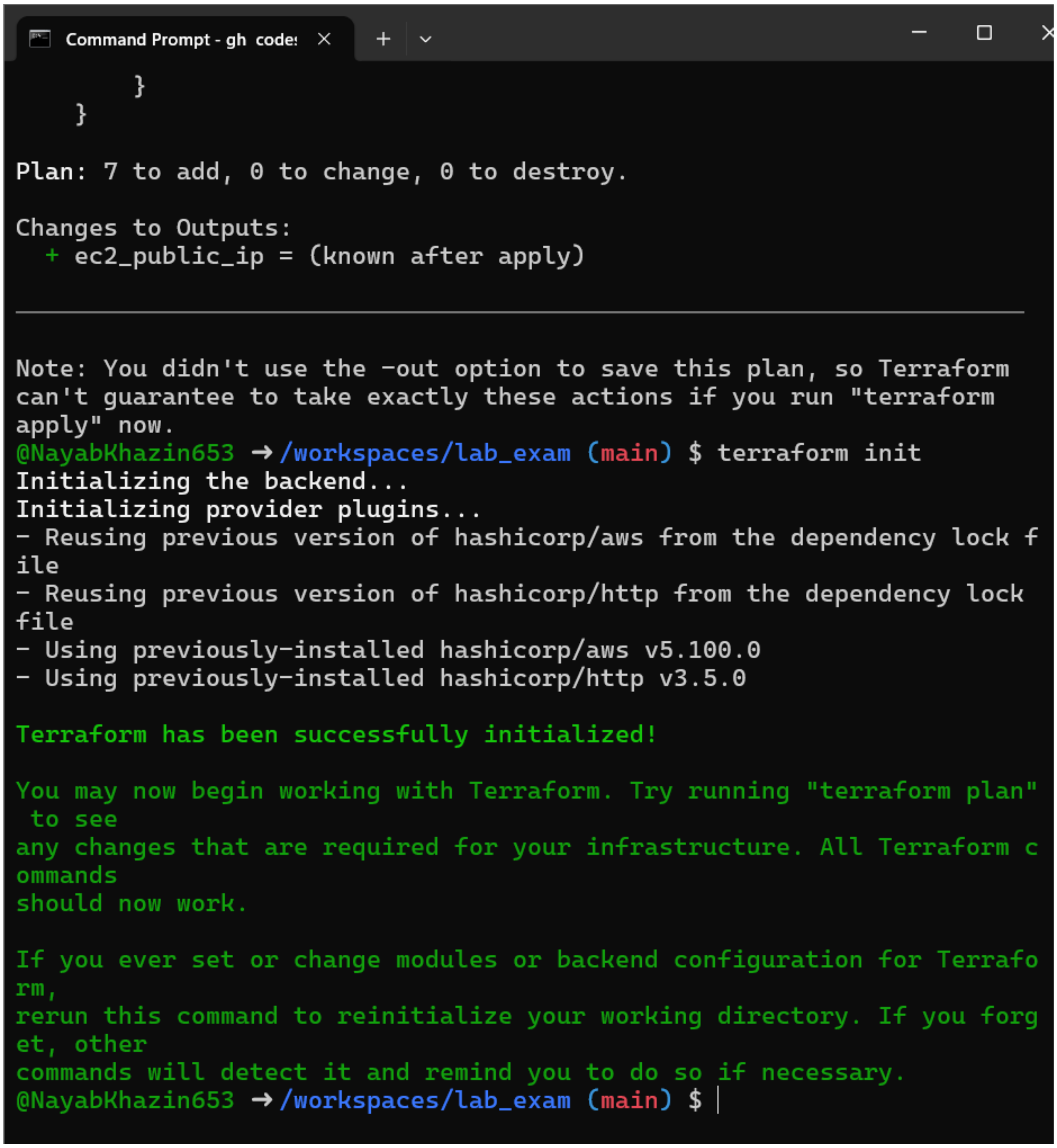
```
@NayabKhazin653 → /workspaces/lab_exam (main) $ cat outputs.tf
output "ec2_public_ip" {
  description = "Public IP of EC2 instance"
  value       = aws_instance.myapp_ec2.public_ip
}
@NayabKhazin653 → /workspaces/lab_exam (main) $
```

q2_output_block.png



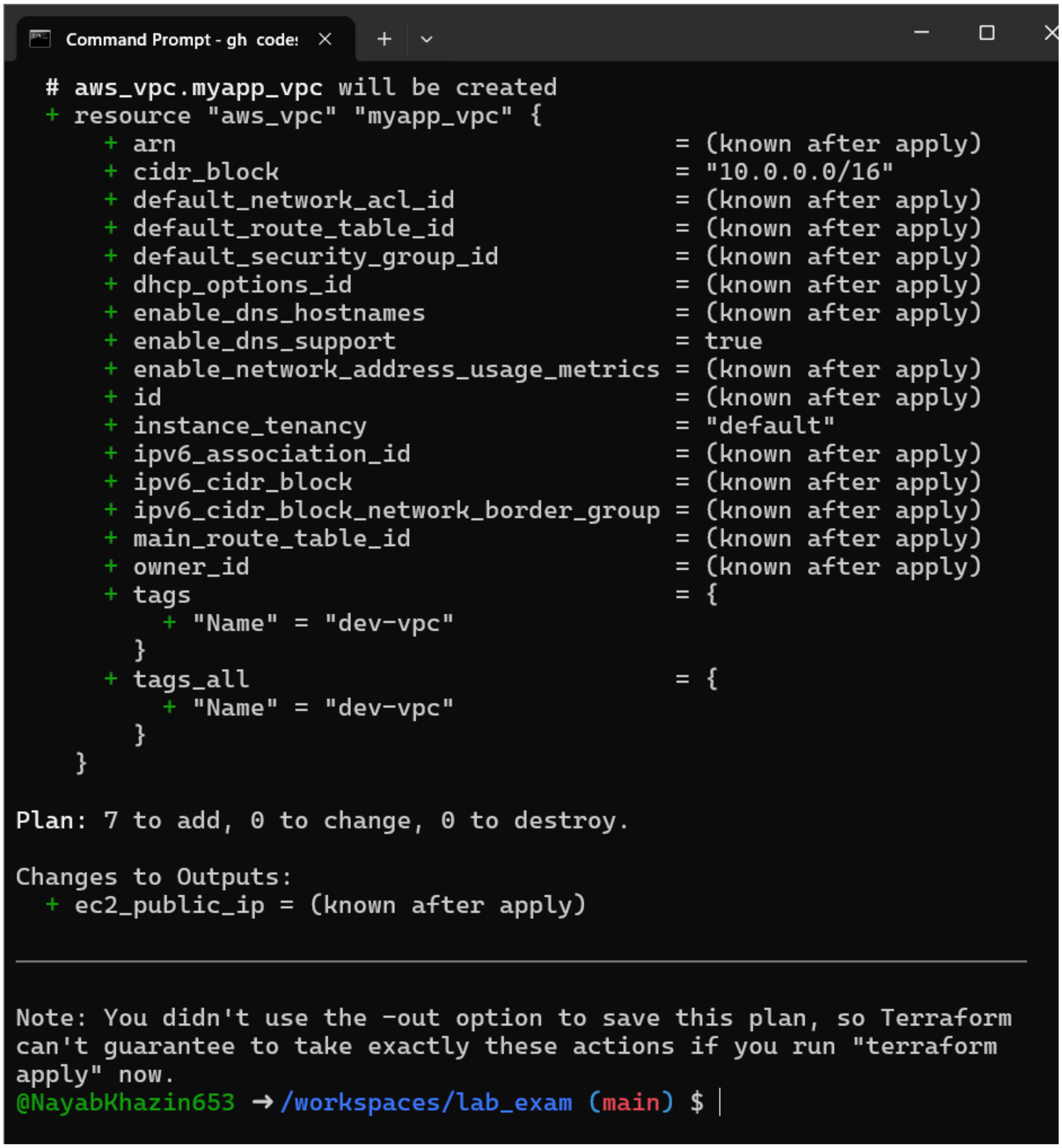
```
@NayabKhazin653 → /workspaces/lab_exam (main) $ cat terraform.tfvars
vpc_cidr_block    = "10.0.0.0/16"
subnet_cidr_block = "10.0.10.0/24"
availability_zone = "me-central-1a"
env_prefix        = "dev"
instance_type     = "t3.micro"
@NayabKhazin653 → /workspaces/lab_exam (main) $
```

q2_tfvars_or_vars.png



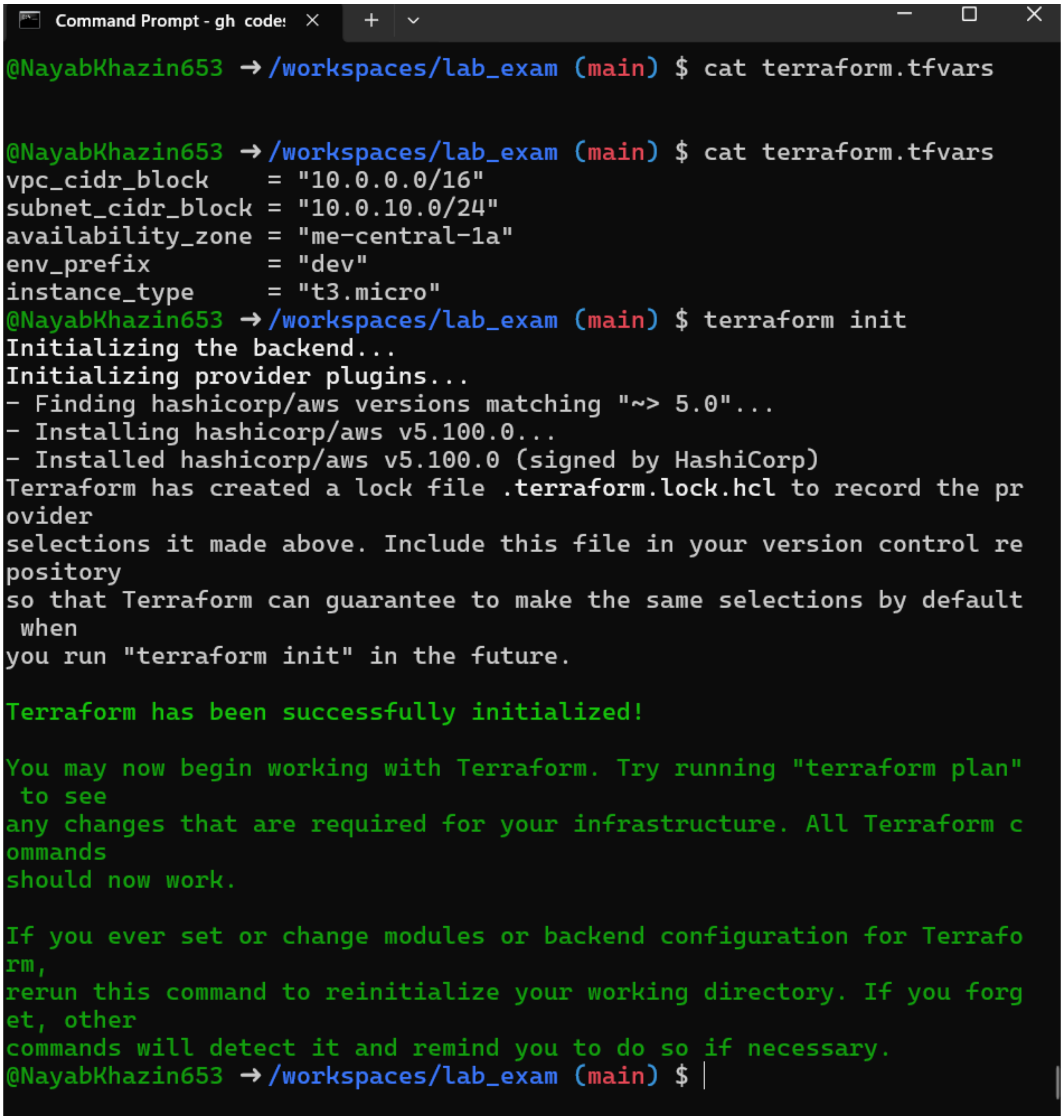
```
        }
    }

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

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

───────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform
can't guarantee to take exactly these actions if you run "terraform
apply" now.
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock f
ile
- Reusing previous version of hashicorp/http from the dependency lock
file
- Using previously-installed hashicorp/aws v5.100.0
- Using previously-installed hashicorp/http v3.5.0

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 c
ommands
should now work.

If you ever set or change modules or backend configuration for Terrafo
rm,
rerun this command to reinitialize your working directory. If you forg
et, other
commands will detect it and remind you to do so if necessary.
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $
```

q2_terraform_init.png



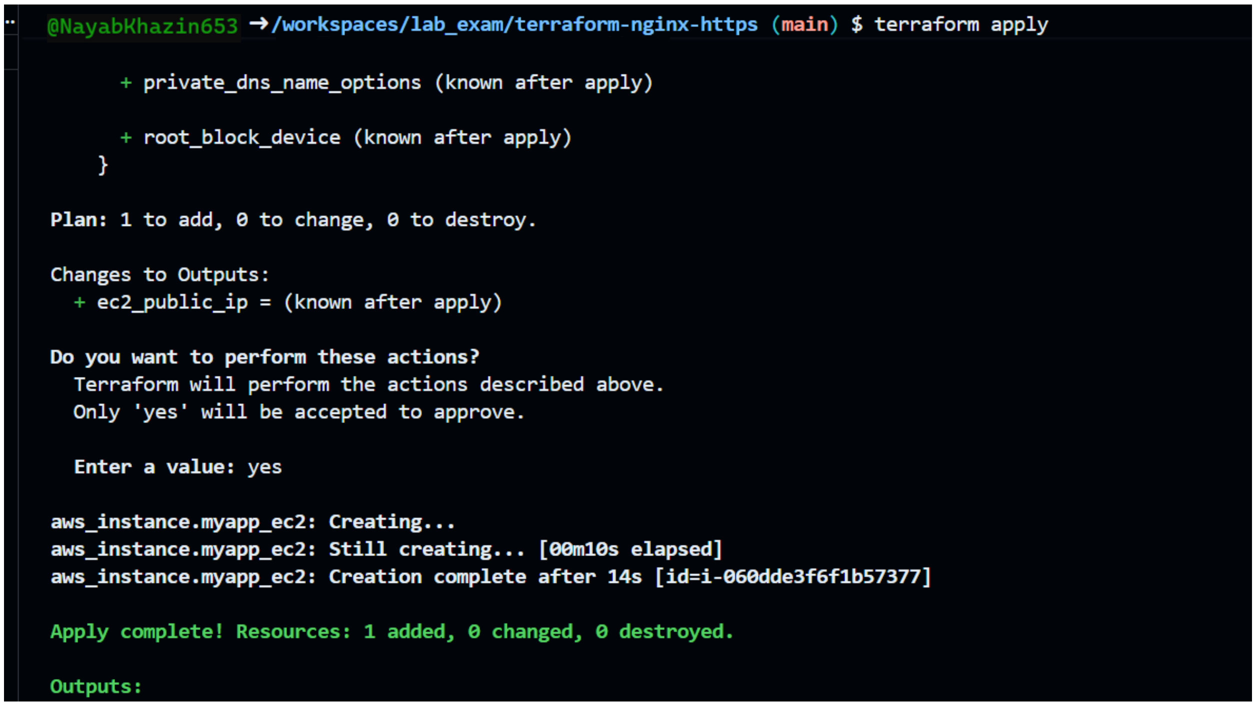
```
  # aws_vpc.myapp_vpc will be created
  + resource "aws_vpc" "myapp_vpc" {
      + arn                                  = (known after apply)
      + cidr_block                           = "10.0.0.0/16"
      + default_network_acl_id               = (known after apply)
      + default_route_table_id               = (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)

─────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform
can't guarantee to take exactly these actions if you run "terraform
apply" now.
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $
```

q2_terraform_plan.png

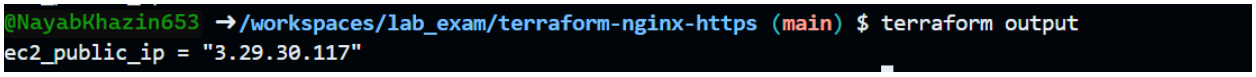


```
Command Prompt - gh  code:
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ cat terraform.tfvars


@NayabKhazin653 ➜ /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"
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Installing hashicorp/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the pr
ovider
selections it made above. Include this file in your version control re
pository
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 c
ommands
should now work.

If you ever set or change modules or backend configuration for Terrafo
rm,
rerun this command to reinitialize your working directory. If you forg
et, other
commands will detect it and remind you to do so if necessary.
@NayabKhazin653 ➜ /workspaces/lab_exam (main) $
```

q2_terraform_init.png



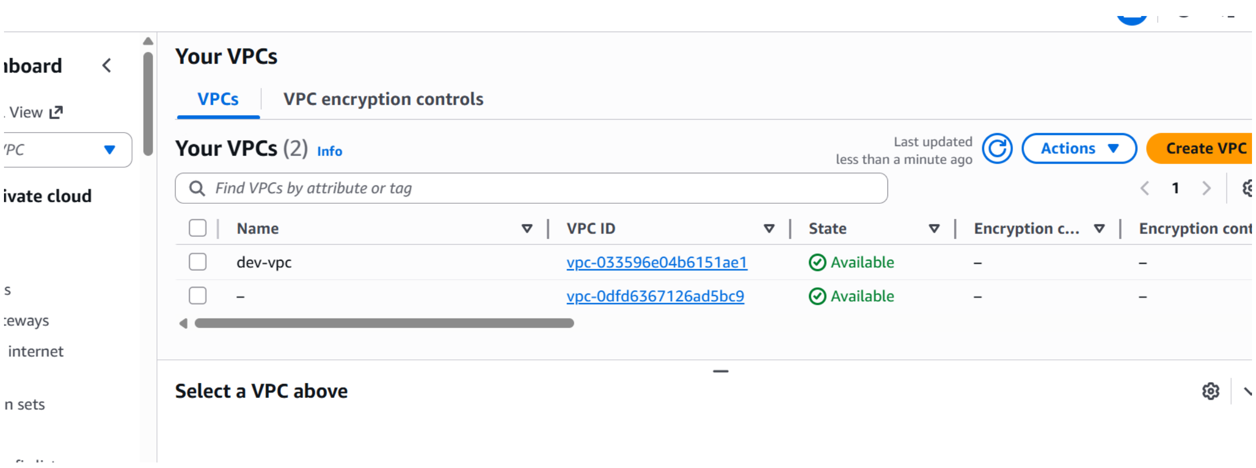
```
@NayabKhazin653 →/workspaces/lab_exam/terraform-nginx-https (main) $ terraform apply

        + 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)

  Do you want to perform these actions?
    Terraform will perform the actions described above.
    Only 'yes' will be accepted to approve.

    Enter a value: yes

  aws_instance.myapp_ec2: Creating...
  aws_instance.myapp_ec2: Still creating... [00m10s elapsed]
  aws_instance.myapp_ec2: Creation complete after 14s [id=i-060dde3f6f1b57377]

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

  Outputs:
```



```
@NayabKhazin653 →/workspaces/lab_exam/terraform-nginx-https (main) $ terraform output
ec2_public_ip = "3.29.30.117"
```

Verify Terraform resources in AWS console

VPC and Subnet



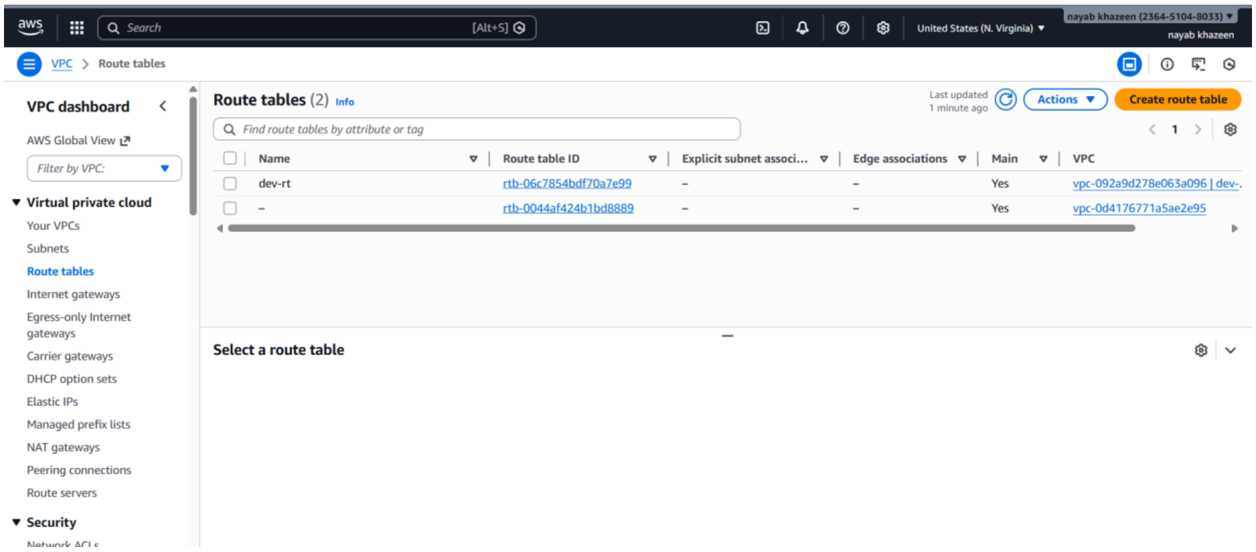
Your VPCs

Your VPCs (2) Info

Last updated less than a minute ago
Actions
Create VPC
Find VPCs by attribute or tag
1

Name	VPC ID	State	Encryption c...	Encryption cont
dev-vpc	vpc-033596e04b6151ae1	Available	–	–
–	vpc-0dfd6367126ad5bc9	Available	–	–

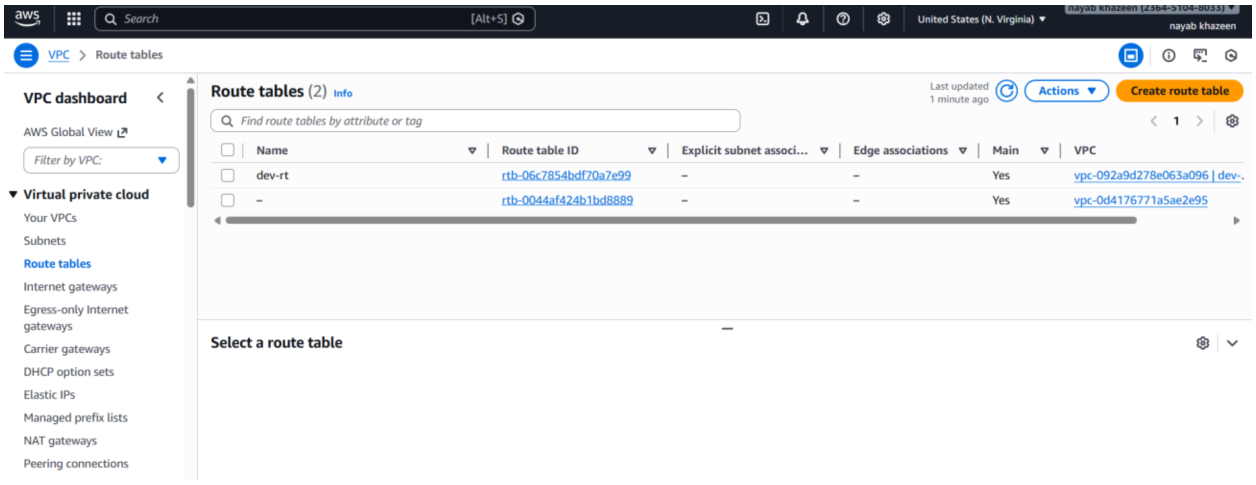
Select a VPC above



Route tables (2) Info

Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
dev-rt	rtb-06c7854bdf70a7e99	–	–	Yes	vpc-092a9d278e063a096 | dev-...
–	rtb-0044af424b1bd8889	–	–	Yes	vpc-0d4176771a5ae2e95

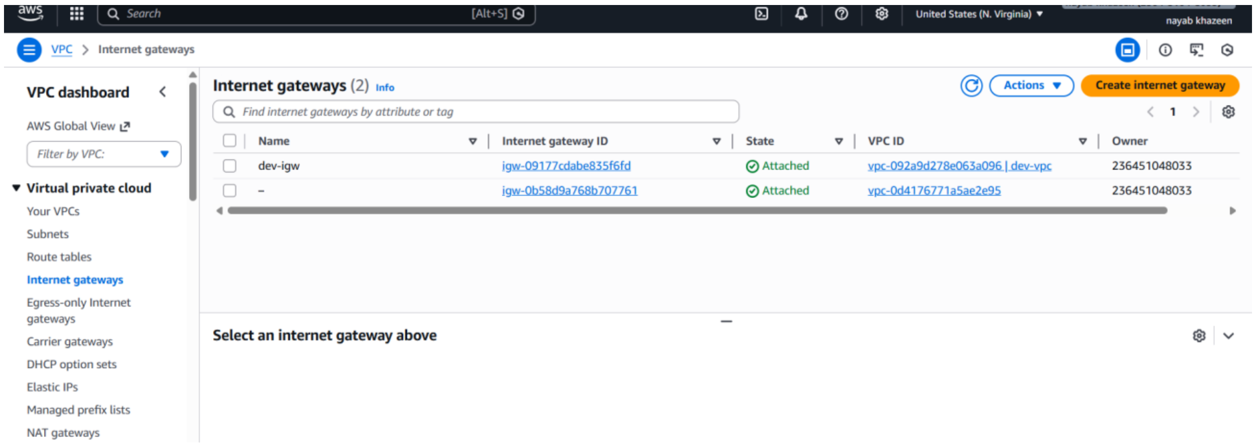
Select a route table



Route tables (2) Info

Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
dev-rt	rtb-06c7854bdf70a7e99	–	–	Yes	vpc-092a9d278e063a096 | dev-...
–	rtb-0044af424b1bd8889	–	–	Yes	vpc-0d4176771a5ae2e95

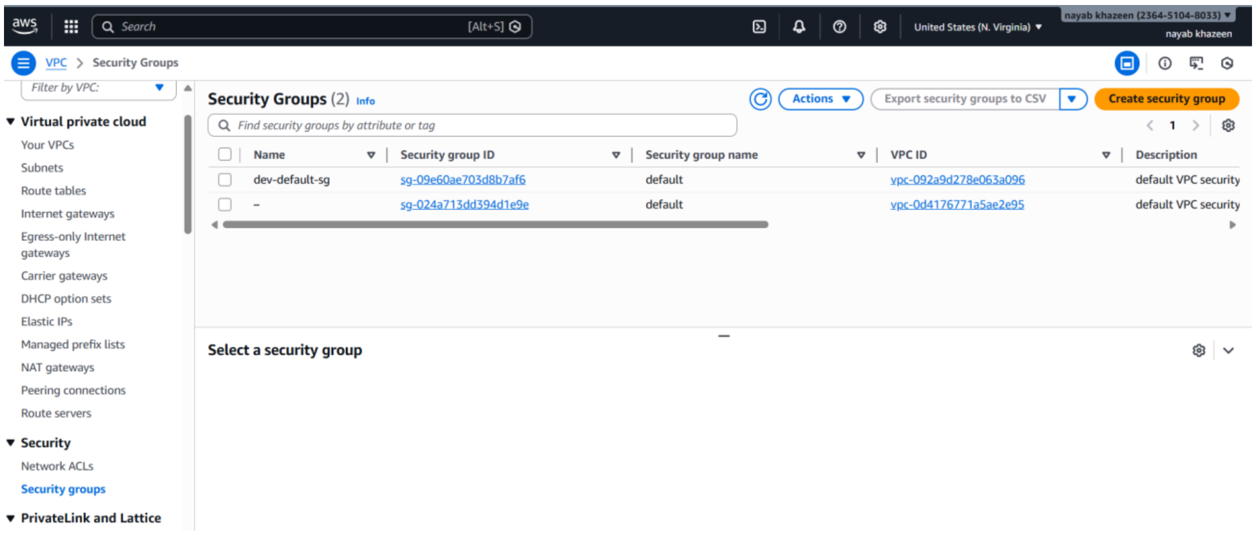
Select a route table



Internet gateways (2) Info

Name	Internet gateway ID	State	VPC ID	Owner
dev-igw	igw-09177cdabe835f6fd	Attached	vpc-092a9d278e063a096 | dev-vpc	236451048033
–	igw-0b58d9a768b707761	Attached	vpc-0d4176771a5ae2e95	236451048033

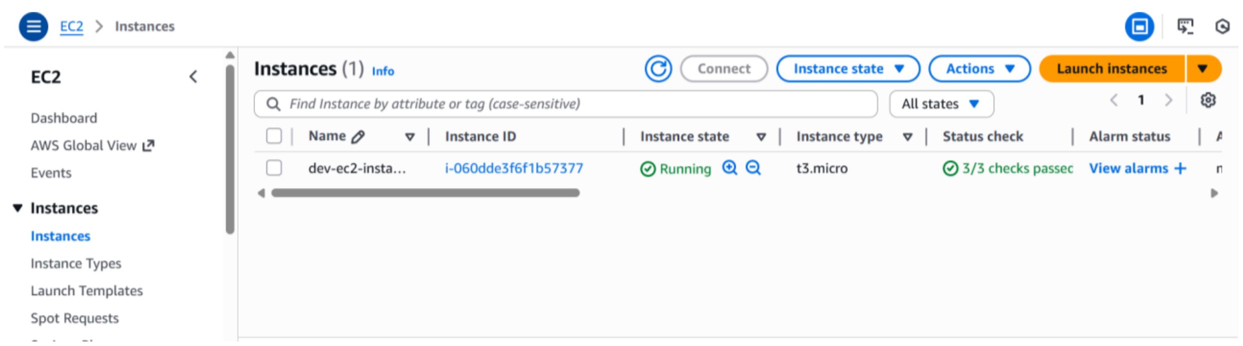
Select an internet gateway above



Security Groups (2) Info

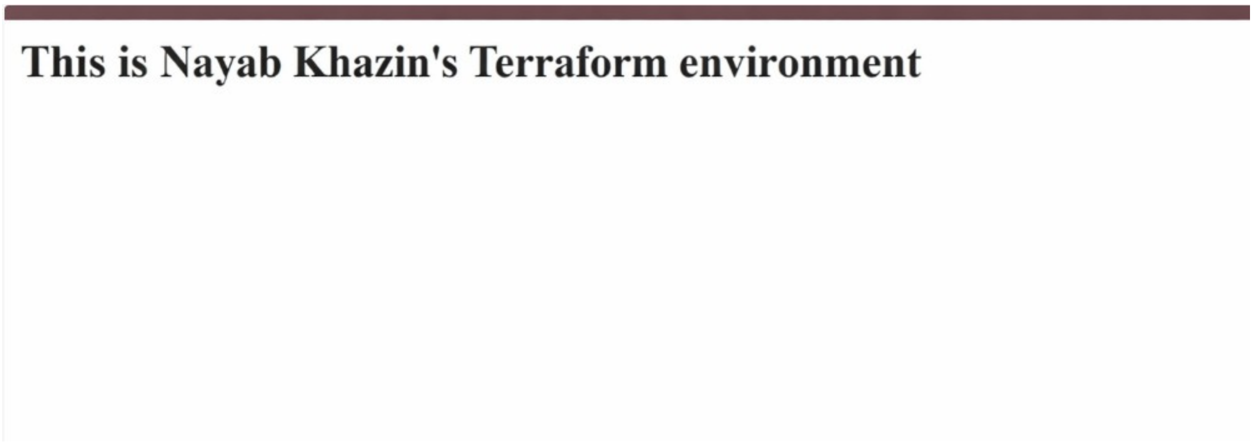
Name	Security group ID	Security group name	VPC ID	Description
dev-default-sg	sg-09e60ae703d8b7af6	default	vpc-092a9d278e063a096	default VPC security
–	sg-024a713dd394d1e9e	default	vpc-0d4176771a5ae2e95	default VPC security

Select a security group



Instances (1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
dev-ec2-insta...	i-060dde3f6f1b57377	Running	t3.micro	3/3 checks passec	View alarms +



This is Nayab Khazin's Terraform environment