Lab Exam

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

Step 1: Create IAM Group

aws iam create-group --group-name SoftwareEngineering

```
@23-22411-029-sudo ➜/workspaces/Exam (main) $ aws iam create-group --group-name SoftwareEngineering
{
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPARPRNTUQVX3N3WLDRN",
        "Arn": "arn:aws:iam::102101132331:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:34:38+00:00"
    }
}
```

aws iam get-group --group-name SoftwareEngineering

```
@23-22411-029-sudo ➜/workspaces/Exam (main) $ aws iam get-group --group-name SoftwareEngineering
{
    "Users": [],
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPARPRNTUQVX3N3WLDRN",
        "Arn": "arn:aws:iam::102101132331:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:34:38+00:00"
    }
}
```

Step 2: Create IAM User Kainat

aws iam create-user --user-name Kainat

```
@23-22411-029-sudo ➜/workspaces/Exam (main) $ aws iam create-user --user-name Kainat
{
    "User": {
        "Path": "/",
        "UserName": "Kainat",
        "UserId": "AIDARPRNTUQVTG4TFM6RH",
        "Arn": "arn:aws:iam::102101132331:user/Kainat",
        "CreateDate": "2026-01-19T07:36:13+00:00"
    }
}
```

aws iam get-user --user-name Kainat

```
@23-22411-029-sudo ➜/workspaces/Exam (main) $ aws iam get-user --user-name Kainat
{
    "User": {
        "Path": "/",
        "UserName": "Kainat",
        "UserId": "AIDARPRNTUQVTG4TFM6RH",
        "Arn": "arn:aws:iam::102101132331:user/Kainat",
        "CreateDate": "2026-01-19T07:36:13+00:00"
    }
}
```

Step 3: Add User to Group

aws iam add-user-to-group \
  --user-name Kainat \
  --group-name SoftwareEngineering

```
@23-22411-029-sudo ➜/workspaces/Exam (main) $ aws iam add-user-to-group \
    --user-name Kainat \
    --group-name SoftwareEngineering
```

aws iam get-group --group-name SoftwareEngineering

```
@23-22411-029-sudo ➜/workspaces/Exam (main) $ aws iam get-group --group-name SoftwareEngineering
{
    "Users": [
        {
            "Path": "/",
            "UserName": "Kainat",
            "UserId": "AIDARPRNTUQVTG4TFM6RH",
            "Arn": "arn:aws:iam::102101132331:user/Kainat",
            "CreateDate": "2026-01-19T07:36:13+00:00"
        }
    ],
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPARPRNTUQVX3N3WLDRN",
        "Arn": "arn:aws:iam::102101132331:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:34:38+00:00"
    }
}
```

Step 4: Find AdministratorAccess Policy

aws iam list-policies --scope AWS | grep AdministratorAccess

```
@23-22411-029-sudo →/workspaces/Exam (main) $ aws iam list-policies --scope AWS | grep AdministratorAccess
            "PolicyName": "AdministratorAccess",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "PolicyName": "AdministratorAccess-Amplify",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess-Amplify",
            "PolicyName": "AWSAuditManagerAdministratorAccess",
            "Arn": "arn:aws:iam::aws:policy/AWSAuditManagerAdministratorAccess",
            "PolicyName": "AdministratorAccess-AWSElasticBeanstalk",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess-AWSElasticBeanstalk",
            "PolicyName": "AWSManagementConsoleAdministratorAccess",
            "Arn": "arn:aws:iam::aws:policy/job-function/AWSManagementConsoleAdministratorAccess",
@23-22411-029-sudo →/workspaces/Exam (main) $
```

Step 5: Attach Policy to Group

aws iam attach-group-policy \
  --group-name SoftwareEngineering \
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess

```
@23-22411-029-sudo →/workspaces/Exam (main) $ aws iam attach-group-policy \
  --group-name SoftwareEngineering \
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
@23-22411-029-sudo →/workspaces/Exam (main) $
```

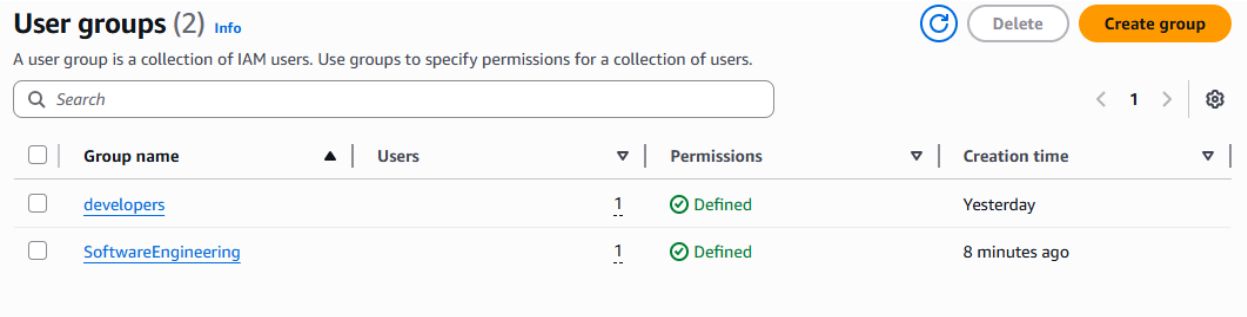
aws iam list-attached-group-policies \
  --group-name SoftwareEngineering

```
@23-22411-029-sudo →/workspaces/Exam (main) $ aws iam list-attached-group-policies \
  --group-name SoftwareEngineering
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        }
    ]
}
@23-22411-029-sudo →/workspaces/Exam (main) $
```

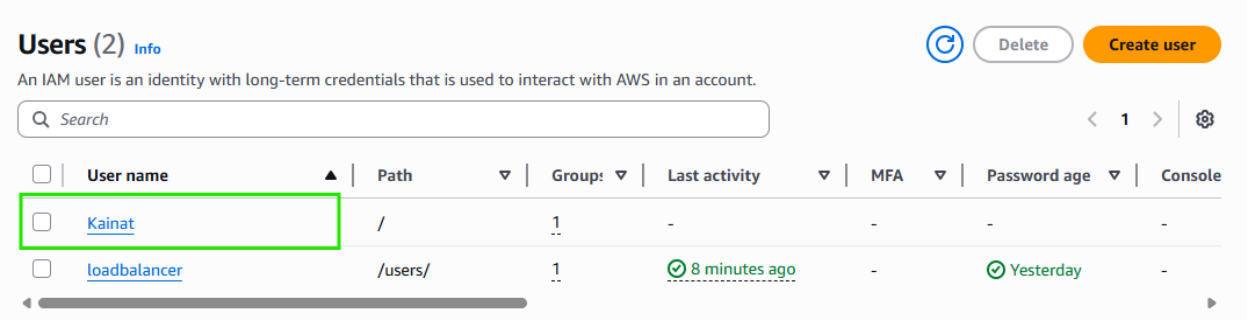
Step 6: Console Verification

Go to **IAM Console**

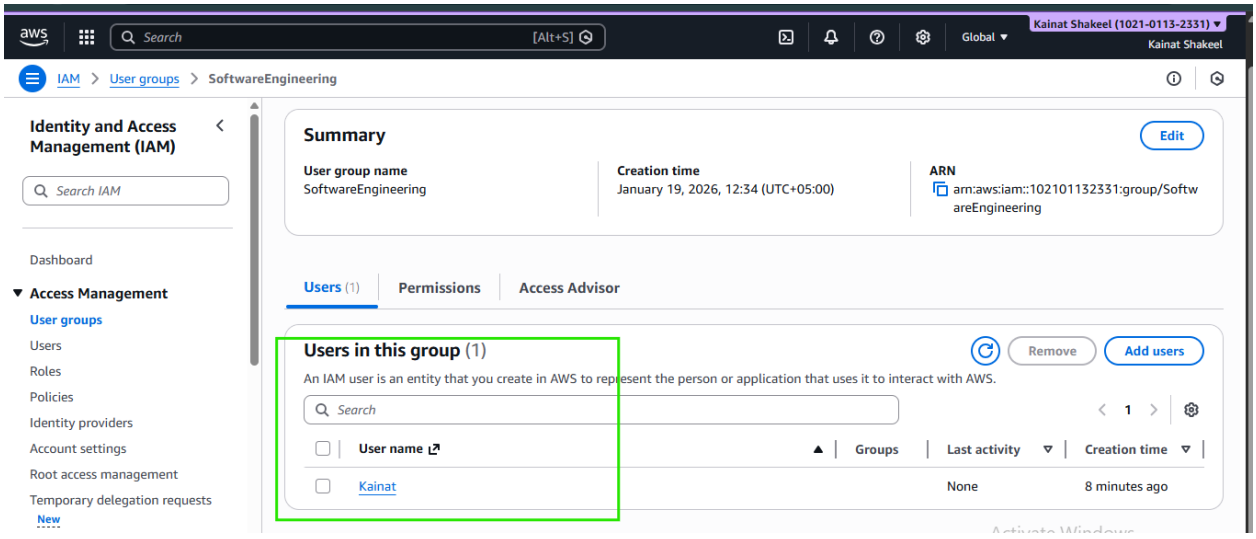
- q1_console_group.png



- q1_console_user_in_group.png



- q1_console_group_policy.png



— TERRAFORM (30 MARKS)

Step 1: Folder Structure

mkdir terraform
cd terraform

```
@23-22411-029-sudo →/workspaces/Exam (main) $ mkdir terraform
cd terraform
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $
```

Files you WILL create:

main.tf
variables.tf
outputs.tf
entry-script.sh
terraform.tfvars

```
@23-22411-029-sudo →/workspaces/Exam (main) $ mkdir terraform
cd terraform
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $ touch main.tf
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $ touch variables.tf
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $ touch outputs.tf
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $ touch entry-script.sh
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $ touch terraform.tfvars
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $ ls
entry-script.sh  main.tf  outputs.tf  terraform.tfvars  variables.tf
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $
```

Step 2: Provider

```
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $ cat ~/.aws/config
[default]
region = ap-south-1
output = json
@23-22411-029-sudo →/workspaces/Exam/terraform (main) $
```

Step 1 – Configure AWS provider

Edit `main.tf` to configure the AWS provider:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

  GNU nano 7.2
provider "aws" {
  region = var.aws_region  # Read region from variable
}
```

Step 2 – Define input variables

Edit **`variables.tf`**:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

  GNU nano 7.2
variable "aws_region" {
  description = "AWS region"
  default     = "ap-south-1"
}

variable "vpc_cidr_block" {
  description = "CIDR block for VPC"
  type        = string
}

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

variable "availability_zone" {

^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Exec
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Just
```

Step 3 – Create VPC and Subnet

Add to **`main.tf`**:

```
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" }
}
```

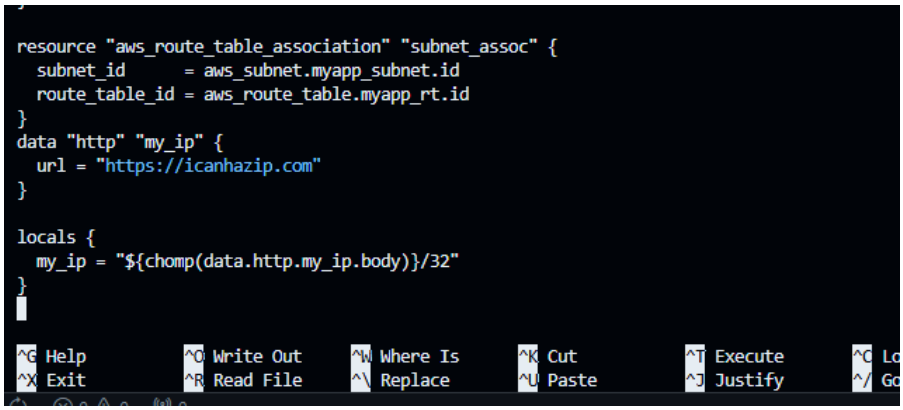
Step 4 – Internet Gateway & Route Table

Add to **`main.tf`**:

```
  GNU nano 7.2                                                  main.tf *
  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" }
}

resource "aws_route_table" "myapp_rt" {
  vpc_id = aws_vpc.myapp_vpc.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.myapp_igw.id
  }
  tags = { Name = "${var.env_prefix}-rt" }
}

resource "aws_route_table_association" "subnet_assoc" {
  subnet_id      = aws_subnet.myapp_subnet.id
  route_table_id = aws_route_table.myapp_rt.id
}
```

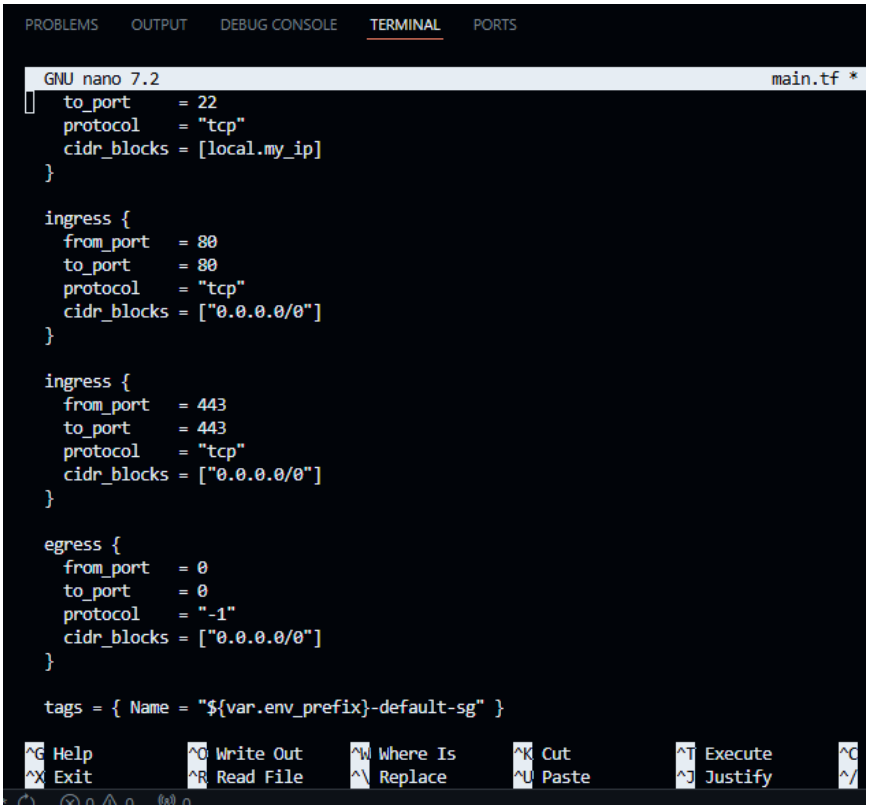
Step 5 – Discover public IP & define local /32

Add to **`main.tf`**:



Step 6 – Default Security Group

Add to **`main.tf`**:



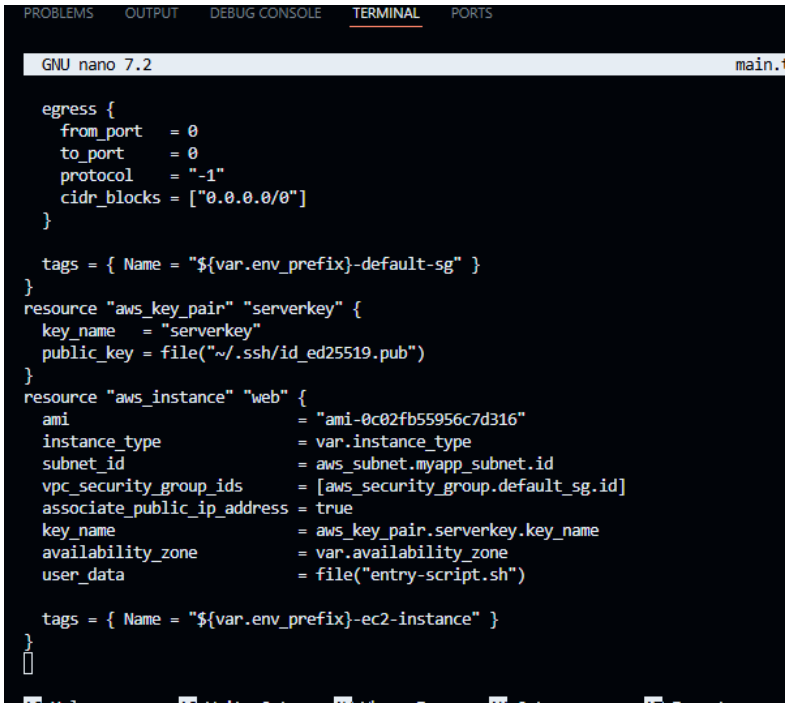
Step 7 – Key Pair

Add to **`main.tf`**:

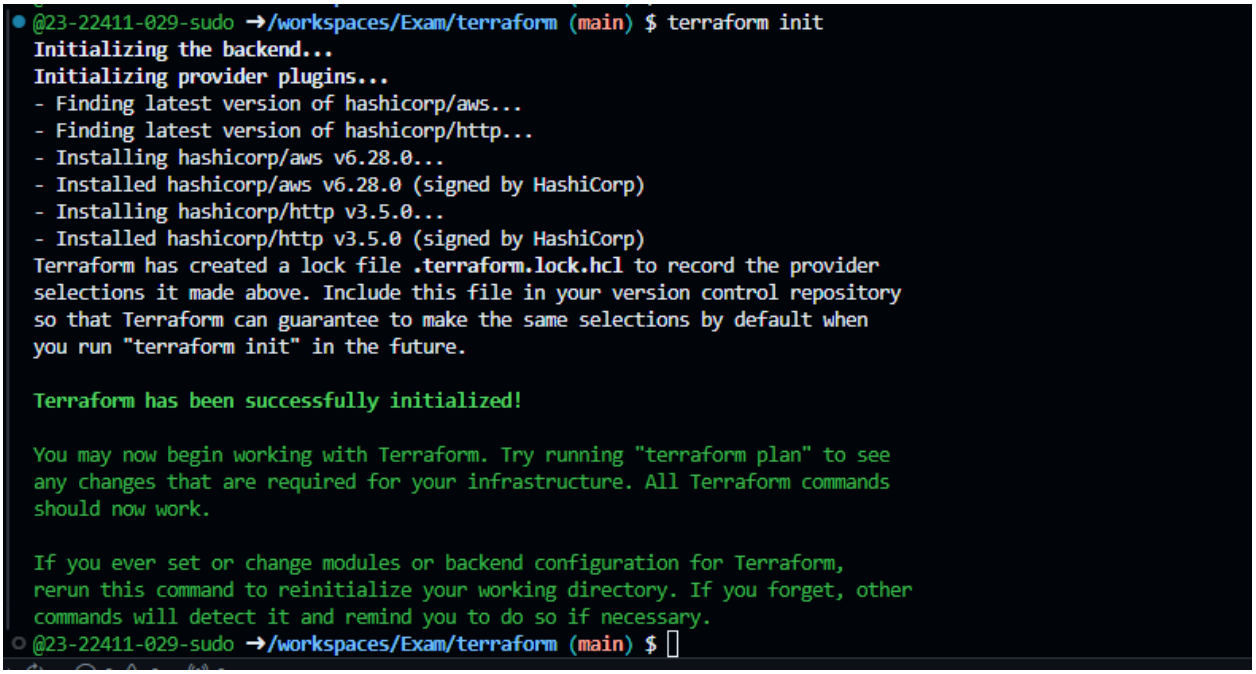
```
  tags = { Name = "${var.env_prefix}-default-sg" }
}
resource "aws_key_pair" "serverkey" {
  key_name   = "serverkey"
  public_key = file("~/.ssh/id_ed25519.pub")
}
```

Step 8 – EC2 Instance

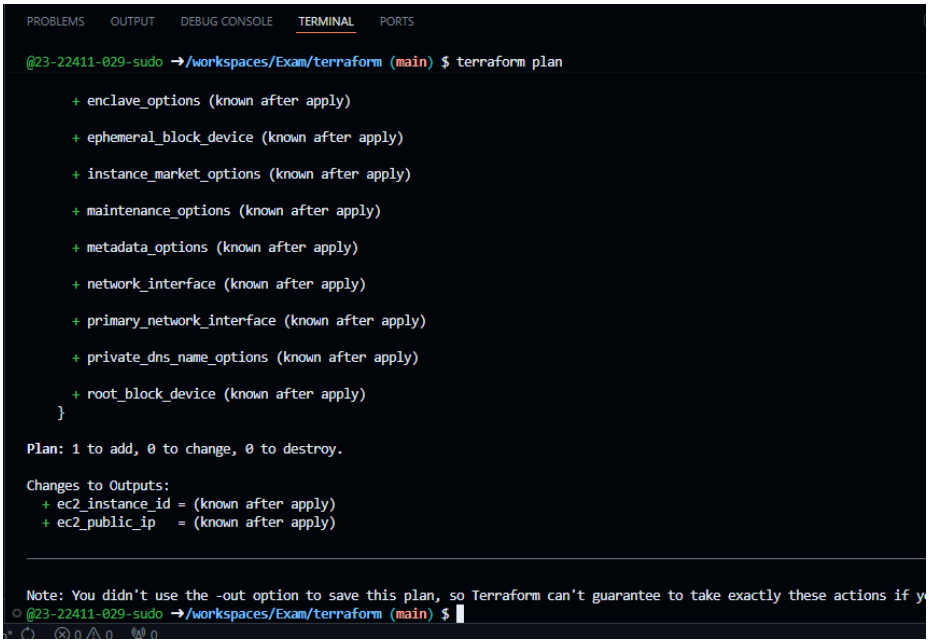
Add to **`main.tf`**:

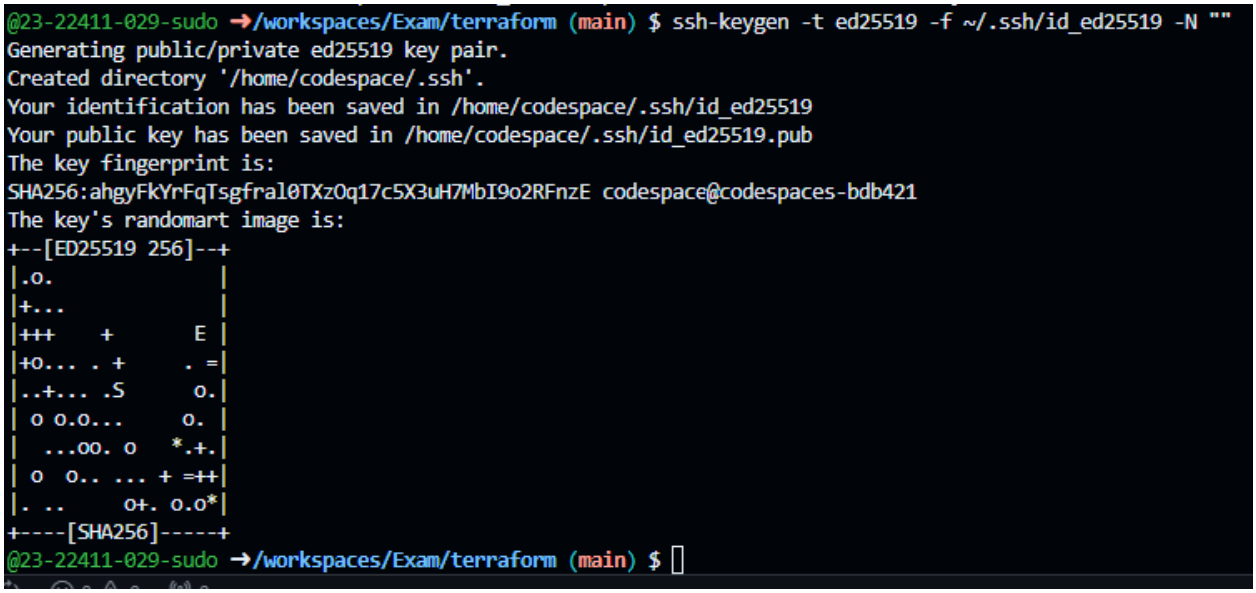


– Run Terraform

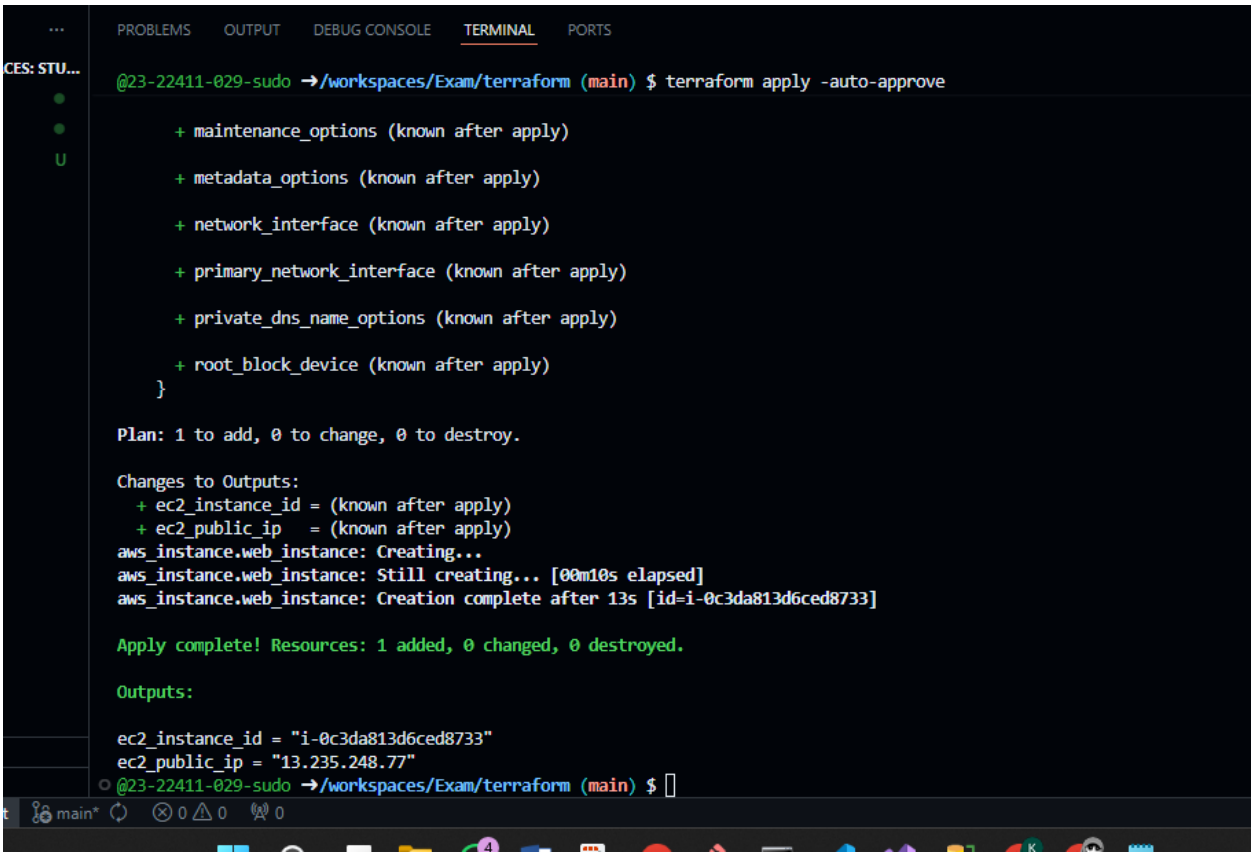


terraform plan

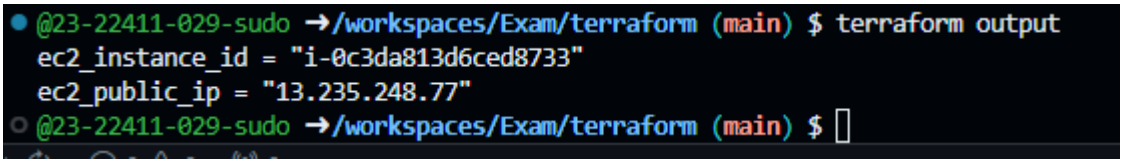




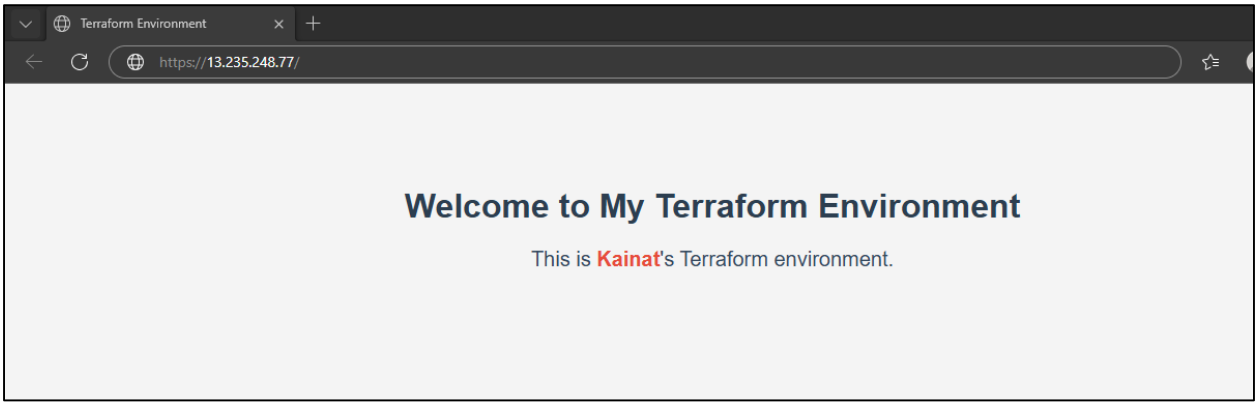
terraform apply -auto-approve



`terraform output`

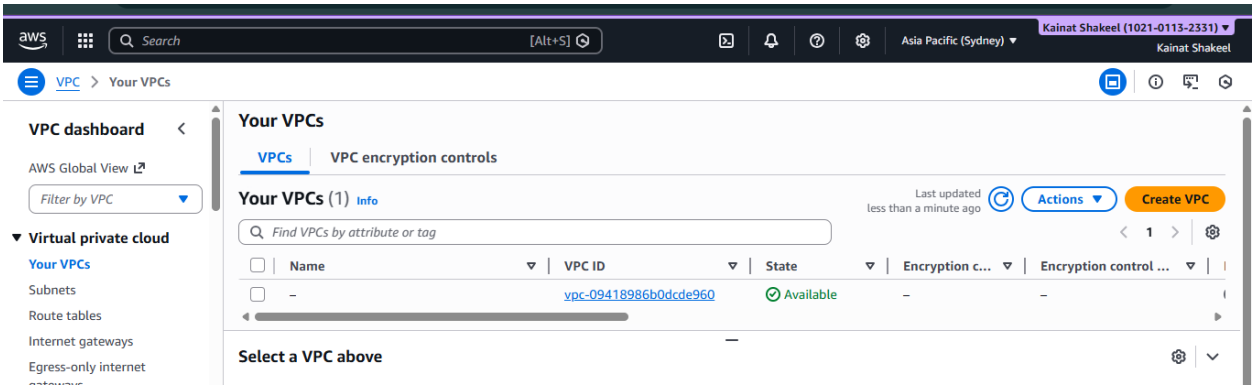


Apply the Terraform configuration

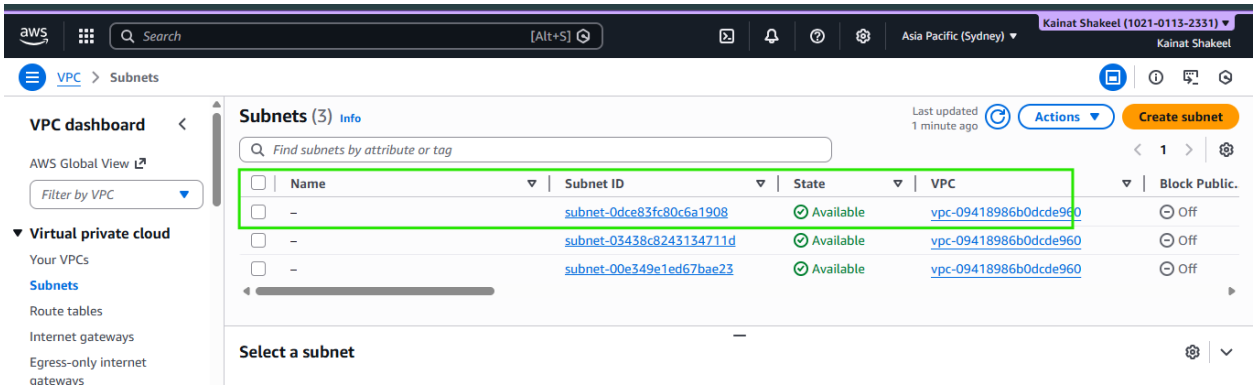


Step 13 – Verify AWS Console

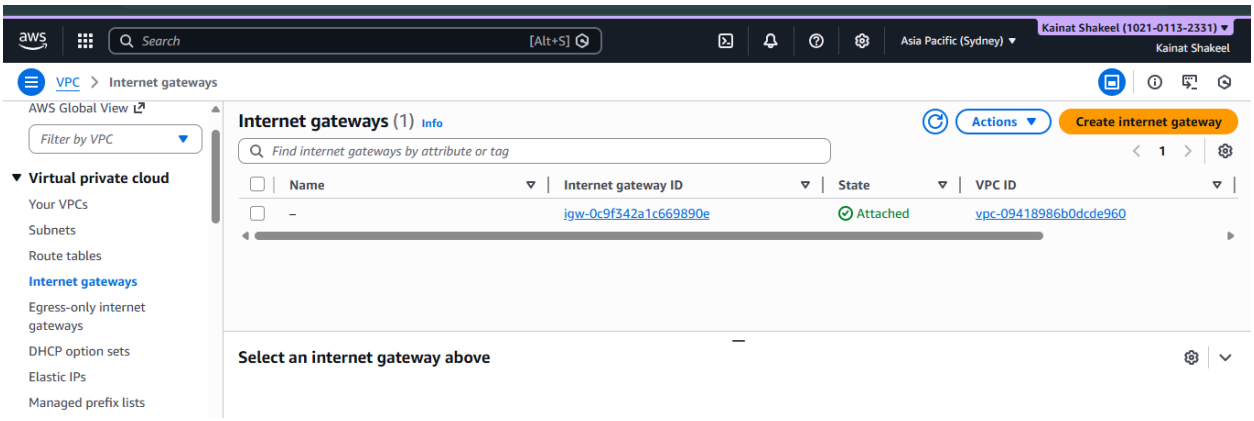
VPC: `q2_console_vpc.png`



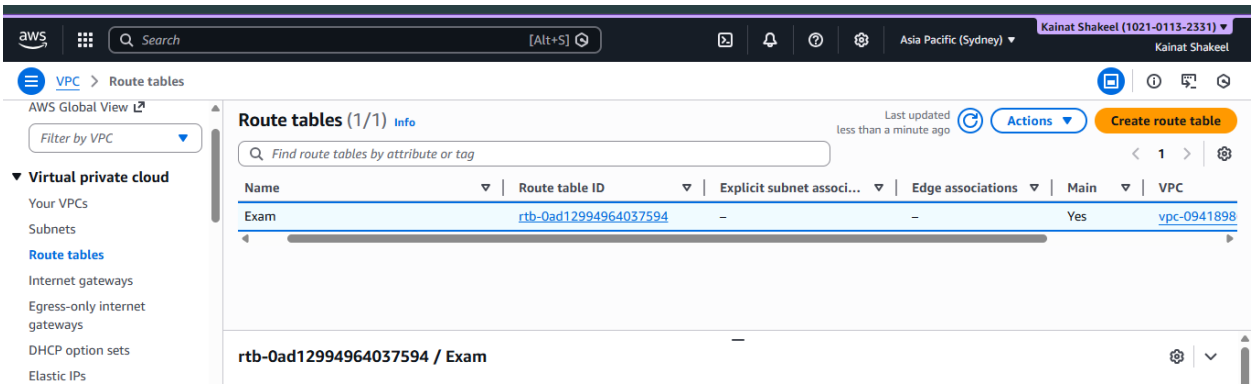
- **Subnet:** `q2_console_subnet.png`



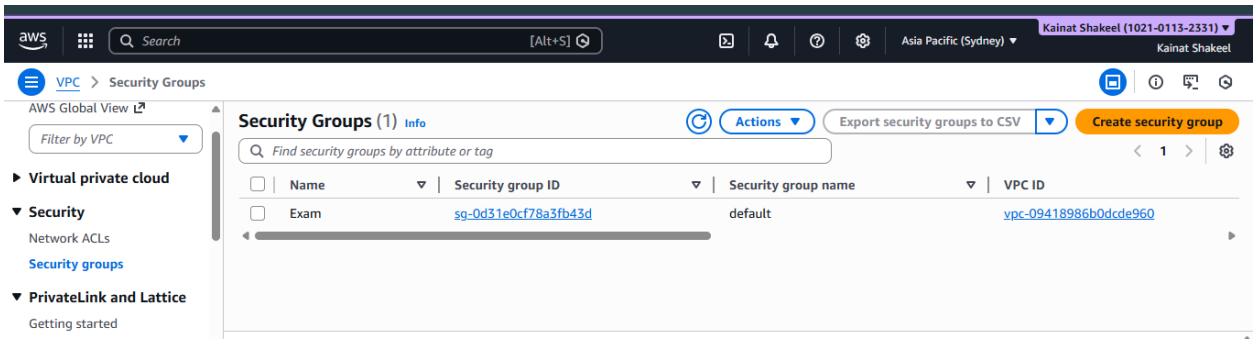
- **IGW:** `q2_console_igw.png`



- **Route Table:** `q2_console_route_table.png`



- **Security Group:** `q2_console_sg.png`



DESTORY

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

@23-22411-029-sudo ➜/workspaces/Exam/terraform (main) $ terraform destroy -auto-approve
    }

Plan: 0 to add, 0 to change, 8 to destroy.

Changes to Outputs:
  - ec2_instance_id = "i-0c3da813d6ced8733" -> null
  - ec2_public_ip   = "13.235.248.77" -> null
aws_route_table_association.subnet_assoc: Destroying... [id=rtbassoc-0ca25b417b7bba26a]
aws_instance.web_instance: Destroying... [id=i-0c3da813d6ced8733]
aws_route_table_association.subnet_assoc: Destruction complete after 1s
aws_route_table.myapp_rt: Destroying... [id=rtb-0d1130ae316da95f7]
aws_route_table.myapp_rt: Destruction complete after 0s
aws_internet_gateway.myapp_igw: Destroying... [id=igw-0e99d01944b9e76f4]
aws_instance.web_instance: Still destroying... [id=i-0c3da813d6ced8733, 00m10s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0e99d01944b9e76f4, 00m10s elapsed]
aws_instance.web_instance: Still destroying... [id=i-0c3da813d6ced8733, 00m20s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0e99d01944b9e76f4, 00m20s elapsed]
aws_internet_gateway.myapp_igw: Destruction complete after 27s
aws_instance.web_instance: Destruction complete after 30s
aws_subnet.myapp_subnet: Destroying... [id=subnet-016bfe32bc1ebb413]
aws_key_pair.serverkey: Destroying... [id=serverkey]
aws_security_group.default_sg: Destroying... [id=sg-04028c763ba9d230f]
aws_key_pair.serverkey: Destruction complete after 0s
aws_subnet.myapp_subnet: Destruction complete after 0s
aws_security_group.default_sg: Destruction complete after 1s
aws_vpc.myapp_vpc: Destroying... [id=vpc-031bc8db74c63a71a]
aws_vpc.myapp_vpc: Destruction complete after 0s

Destroy complete! Resources: 8 destroyed.
@23-22411-029-sudo ➜/workspaces/Exam/terraform (main) $
```

Question#3

Step 1: Create the Ansible inventory file (`hosts`)

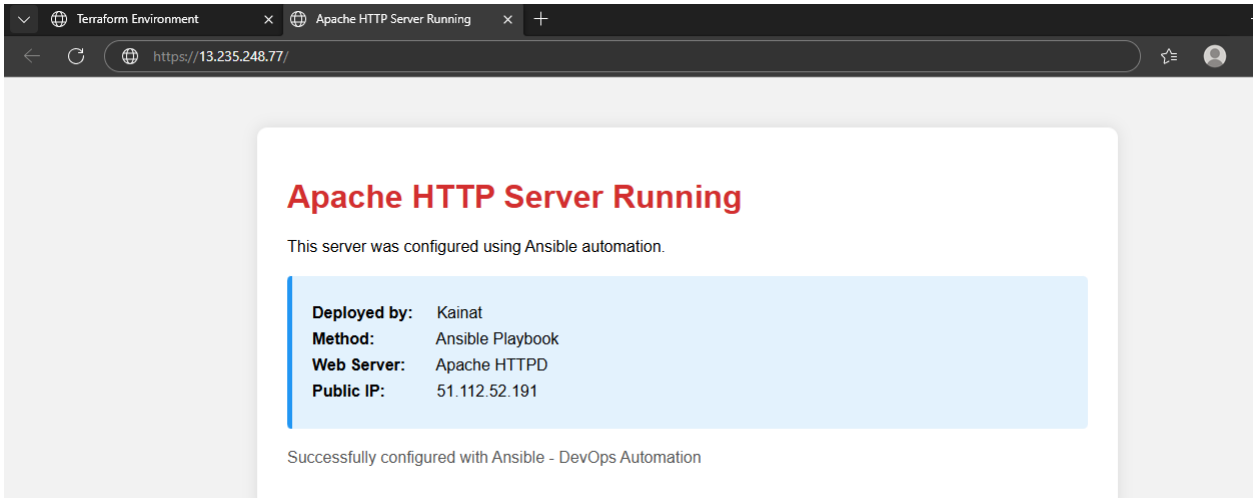
```
Destroy complete! Resources: 8 destroyed.
@23-22411-029-sudo ➜/workspaces/Exam/terraform (main) $ mkdir -p ~/Lab_exam/ansible
cd ~/Lab_exam/ansible
@23-22411-029-sudo ➜~/Lab_exam/ansible $ nano hosts
@23-22411-029-sudo ➜~/Lab_exam/ansible $
```

```
@23-22411-029-sudo ➜~/Lab_exam/ansible $ timeout 5 curl http://https://13.235.248.77/ 2>&
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1802  100  1802    0     0   8352      0 --:--:-- --:--:-- --:--:--  8342
<!DOCTYPE html>
<html>
<head>
    <title>Terraform Environment</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 50px;
            background-color: #f5f5f5;
        }
        .container {
            background-color: white;
            padding: 30px;
            border-radius: 8px;
            box-shadow: 0 2px 4px rgba(0,0,0,0.1);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
 GNU nano 7.2                                                  my-playbook.yml *
        X-aws-ec2-metadata-token-ttl-seconds: "21600"
      register: imds_token

    - name: Get public IPv4 using IMDSv2
      uri:
        url: http://169.254.169.254/latest/meta-data/public-ipv4
        method: GET
        headers:
          X-aws-ec2-metadata-token: "{{ imds_token.content }}"
      register: public_ip

    - name: Get public hostname using IMDSv2
      uri:
        url: http://169.254.169.254
```

```
ybook.yml
ok: [51.112.52.191]

TASK [Print public IP address] ************************************************
ok: [51.112.52.191] =>
  msg:
  - 'Public IP: 51.112.52.191'
  - 'Public Hostname: '

TASK [Restart httpd service] **************************************************
changed: [51.112.52.191]

TASK [Verify httpd is running] ************************************************
ok: [51.112.52.191]

TASK [Display httpd status] ***************************************************
ok: [51.112.52.191] =>
  msg: Apache HTTPD is active

PLAY RECAP ********************************************************************
51.112.52.191              : ok=14   changed=5    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```


Terraform Environment
Apache HTTP Server Running
https://13.235.248.77/
Apache HTTP Server Running
This server was configured using Ansible automation.
Deployed by: Kainat
Method: Ansible Playbook
Web Server: Apache HTTPD
Public IP: 51.112.52.191
Successfully configured with Ansible - DevOps Automation

****