

Zaheena Anwar

2023-Bse-070

BSE-5B

Question 1

```
@Zaheena1 → /workspaces/Lab_exam (main) $ aws iam create-group --group-name SoftwareEngineering
~
(END)...skipping...
{
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPA35PWBHQSORESCEUKC",
        "Arn": "arn:aws:iam::819244121124:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:50:52+00:00"
    }
}
~
~
~
~
~

@Zaheena1 ↵ /workspaces/Lab_exam (main) $ aws iam get-group --group-name SoftwareEngineering
{
    "Users": [],
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPA35PWBHQSORESCEUKC",
        "Arn": "arn:aws:iam::819244121124:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:50:52+00:00"
    }
}
@Zaheena1 ↵ /workspaces/Lab_exam (main) $
```

```
[+] An error occurred (EntityAlreadyExists) when calling the CreateUser operation: user with name Zahina  
@Zaheena1 ② /workspaces/Lab_exam (main) $ aws iam create-user --user-name Zahina  
{  
    "User": {  
        "Path": "/",  
        "UserName": "Zahina",  
        "UserId": "AIDA35PWBHQSDT6RM3BRM",  
        "Arn": "arn:aws:iam::819244121124:user/Zahina",  
        "CreateDate": "2026-01-19T08:02:52+00:00"  
    }  
}  
@Zaheena1 ② /workspaces/Lab_exam (main) $
```

```
[+] @Zaheena1 ② /workspaces/Lab_exam (main) $ aws iam get-user --user-name Zahina  
{  
    "User": {  
        "Path": "/",  
        "UserName": "Zahina",  
        "UserId": "AIDA35PWBHQSDT6RM3BRM",  
        "Arn": "arn:aws:iam::819244121124:user/Zahina",  
        "CreateDate": "2026-01-19T08:02:52+00:00"  
    }  
}  
@Zaheena1 ② /workspaces/Lab_exam (main) $
```

```
[+] @Zaheena1 ② /workspaces/Lab_exam (main) $ aws iam add-user-to-group --user-name Zahina --group-name SoftwareEngineering  
@Zaheena1 ② /workspaces/Lab_exam (main) $  
  
[+] @Zaheena1 ② /workspaces/Lab_exam (main) $ aws iam get-group --group-name SoftwareEngineering  
{  
    "Users": [  
        {  
            "Path": "/",  
            "UserName": "Zahina",  
            "UserId": "AIDA35PWBHQSDT6RM3BRM",  
            "Arn": "arn:aws:iam::819244121124:user/Zahina",  
            "CreateDate": "2026-01-19T08:02:52+00:00"  
        }  
    ],  
    "Group": {  
        "Path": "/",  
        "GroupName": "SoftwareEngineering",  
        "GroupId": "AGPA35PWBHQSORESCEUKC",  
        "Arn": "arn:aws:iam::819244121124:group/SoftwareEngineering",  
        "CreateDate": "2026-01-19T07:50:52+00:00"  
    }  
}  
@Zaheena1 ② /workspaces/Lab_exam (main) $
```

```
[+] @Zaheena1 ② /workspaces/Lab_exam (main) $ aws iam list-policies --query "Policies[?PolicyName=='AdministratorAccess'].{PolicyName:PolicyName, Arn:Arn}" --output table  
+-----+-----+  
| ListPolicies |  
+-----+-----+  
| Arn | PolicyName |  
+-----+-----+  
| arn:aws:iam::aws:policy/AdministratorAccess | AdministratorAccess |  
+-----+-----+  
@Zaheena1 ② /workspaces/Lab_exam (main) $
```

```
[+] @Zaheena1 ② /workspaces/Lab_exam (main) $ aws iam attach-group-policy --group-name SoftwareEngineering --policy-arm arn:aws:iam::aws:policy/AdministratorAccess  
@Zaheena1 ② /workspaces/Lab_exam (main) $ S
```

```
@Zaheena1 ~ /workspaces/Lab_exam (main) $ aws iam list-attached-group-policies --group-name SoftwareEngineering
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        }
    ]
}
@Zaheena1 ~ /workspaces/Lab_exam (main) $
```

The screenshot shows the AWS IAM User Groups page. The left sidebar is titled 'Identity and Access Management (IAM)' and includes 'User groups' (selected), 'Users', 'Roles', 'Policies', 'Identity providers', 'Account settings', 'Root access management', and 'Temporary delegation requests'. The main content area is titled 'SoftwareEngineering' and shows the 'Summary' tab. It displays the user group name 'SoftwareEngineering', creation time 'January 19, 2026, 12:50 (UTC+05:00)', and ARN 'arn:aws:iam:819244121124:group/SoftwareEngineering'. Below this, the 'Users (1)' tab is selected, showing a single user named 'Admin'.

The screenshot shows the AWS IAM Users page. The left sidebar is identical to the previous screen. The main content area is titled 'Users (3)' and shows three users: 'Admin', 'Zaheena', and 'Zahina'. Each user has a status indicator (green circle with a checkmark) and a timestamp ('3 minutes ago', '28 minutes ago', and '1').

The screenshot shows the AWS IAM User details page for 'Zahina'. The left sidebar is identical. The main content area is titled 'Permissions policies (1)' and shows a single policy named 'AdministratorAccess' attached via the 'SoftwareEngineering' group. It also includes sections for 'Permissions boundary (not set)' and 'Generate policy based on CloudTrail events'.

Question 2

```
@Zaheena1 ✘ /workspaces/Lab_exam/lab_q2 (main) $ cat main.tf
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

provider "aws" {
  region           = "us-east-1"
  shared_config_files = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
  profile          = "default"
}
```

```
resource "aws_vpc" "myapp_vpc" {
  cidr_block = var.vpc_cidr_block
  tags = { Name = "${var.env_prefix}-vpc" }
}

resource "aws_subnet" "myapp_subnet_1" {
  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" }
}

resource "aws_default_route_table" "main_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" "myip" {
  url = "https://icanhazip.com"
}

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

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

  ingress {
    from_port = 22
    to_port = 22
    protocol = "tcp"
    cidr_blocks = [local.my_ip]
  }
  ingress {
    from_port = 80
    to_port = 80
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    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" }
}

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

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

```
resource "aws_instance" "myapp_server" {
  ami                      = "ami-04b70fa74e45c3917"
  instance_type             = var.instance_type
  subnet_id                 = aws_subnet.myapp_subnet_1.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.ssh_key.key_name
  user_data                 = file("entry-script.sh")

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

@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $
```

```
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ cat variables.tf
variable "vpc_cidr_block" {}
variable "subnet_cidr_block" {}
variable "availability_zone" {}
variable "env_prefix" {}
variable "instance_type" {}
EOFcat <<EOF > variables.tf
variable "vpc_cidr_block" {}
variable "subnet_cidr_block" {}
variable "availability_zone" {}
variable "env_prefix" {}
variable "instance_type" {}
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ _
```

```
variable "instance_type" {}

@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ cat entry-script.sh
#!/bin/bash
sudo yum update -y
sudo yum install -y nginx mod_ssl

# Create a custom index page with your name
echo "<h1>Hello, this is Urooj's Terraform Environment</h1>" | sudo tee /usr/share/nginx/html/index.html

# Generate Self-Signed Certificate
sudo openssl req -x509 -nodes 365 -newkey rsa:2048 \
  -keyout /etc/pki/tls/private/localhost.key \
  -out /etc/pki/tls/certs/localhost.crt \
  -subj "/C=US/ST=State/L=City/O=Organization/CN=localhost"

# Configure Nginx for HTTPS
sudo cat <<EOC > /etc/nginx/conf.d/ssl.conf
server {
  listen 443 ssl;
  server_name localhost;

  ssl_certificate /etc/pki/tls/certs/localhost.crt;
  ssl_certificate_key /etc/pki/tls/private/localhost.key;

  location / {
    root /usr/share/nginx/html;
    index index.html index.htm;
  }
}
server {
  listen 80;
  server_name localhost;
  return 301 https://\$host\$request_uri;
}
EOC

# Start Nginx
sudo systemctl enable nginx
sudo systemctl start nginx
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ _
```

```
sudo systemctl start nginx
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ cat outputs.tf
output "ec2_public_ip" {
  value = aws_instance.myapp_server.public_ip
}
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $
```

```
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ cat terraform.tfvars
vpc_cidr_block      = "10.0.0.0/16"
subnet_cidr_block   = "10.0.10.0/24"
availability_zone   = "us-east-1a"
env_prefix          = "dev"
instance_type        = "t3.micro"
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $
```

```
  ↵
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (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.
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $
```

```
  ↵
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ terraform plan
data.http.myip: Reading...
data.http.myip: 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.main_rt will be created
+ resource "aws_default_route_table" "main_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
    ]
}
```

```

@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ terraform apply --auto-approve
data.http.myip: Reading...
data.http.myip: 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.main_rt will be created
+ resource "aws_default_route_table" "main_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       = [
}

```

```

@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ terraform output
ec2_public_ip = "100.31.242.176"
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ terraform output
ec2_public_ip = "100.31.242.176"
@Zaheena1 ② /workspaces/Lab_exam/lab_q2 (main) $ terraform

```

The screenshot shows the AWS VPC dashboard. On the left, there's a sidebar with options like 'VPC dashboard', 'AWS Global View', 'Virtual private cloud', and 'Your VPCs'. The main area is titled 'Your VPCs' and shows one entry: 'dev_vpc'. The table details the following:

	Name	VPC ID	State	Block Public Access	DNS hostnames
<input checked="" type="checkbox"/>	dev_vpc	vpc-06798893530e64caf	Available	Off	Enabled

Below the table, there's a summary for 'vpc-06798893530e64caf / dev_vpc' with fields: Details (VPC ID, State, DNS resolution, Tenancy), Block Public Access (Off), DHCP option set (dopt-0ebf8829acde8a637), and Main route table (rtb-0331e1529e5a7cbea).

VPC dashboard

AWS Global View ▾

Filter by VPC ▾

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

DHCP option sets

Elastic IPs

Managed prefix lists

NAT gateways

Peering connections

Subnets (1/3) Info

Last updated less than a minute ago

Actions Create subnet

Name	Subnet ID	State	VPC
-	subnet-05d7c145d3233edc4	Available	vpc-06798893530e64caf dev_...
-	subnet-0323512185672495a	Available	vpc-06798893530e64caf dev_...
<input checked="" type="checkbox"/> dev-subnet-1	subnet-0a8e49be3501d1f24	Available	vpc-06798893530e64caf dev_...

subnet-0a8e49be3501d1f24 / dev-subnet-1

Details Flow logs Route table Network ACL CIDR reservations Sharing Tags

Details

Subnet ID subnet-0a8e49be3501d1f24	Subnet ARN arn:aws:ec2:eu-north-1:819244121124:subnet/subnet-0	State Available	Block Public Access Off
---------------------------------------	---	--------------------	----------------------------

VPC dashboard

AWS Global View ▾

Filter by VPC ▾

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

DHCP option sets

Elastic IPs

Managed prefix lists

NAT gateways

Peering connections

Internet gateways (1/1) Info

Last updated less than a minute ago

Actions Create internet gateway

Name	Internet gateway ID	State	VPC ID
<input checked="" type="checkbox"/> dev-igw	igw-0e439afff8666c275	Attached	vpc-06798893530e64caf dev_vpc

igw-0e439afff8666c275 / dev-igw

Details Tags

Details

Internet gateway ID igw-0e439afff8666c275	State Attached	VPC ID vpc-06798893530e64caf dev_vpc	Owner 819244121124
--	-------------------	---	-----------------------

VPC dashboard

AWS Global View ▾

Filter by VPC ▾

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

DHCP option sets

Elastic IPs

Managed prefix lists

NAT gateways

Peering connections

Route tables (1/1) Info

Last updated less than a minute ago

Actions Create route table

Name	Route table ID	Explicit subnet associations	Edge associations	Main
<input checked="" type="checkbox"/> dev-rt	rtb-0331e1529e5a7cbea	-	-	Yes

rtb-0331e1529e5a7cbea / dev-rt

Details Routes Subnet associations Edge associations Route propagation Tags

Details

Route table ID rtb-0331e1529e5a7cbea	Main Yes	Explicit subnet associations -	Edge associations -
---	-------------	-----------------------------------	------------------------

The screenshot shows two AWS management console pages:

- VPC > Security Groups**: Displays a list of security groups. One group, "dev-default-sg" (ID: sg-03df368a62ee282a4), is selected. Its details show it's a "default" security group for a VPC with ID vpc-06798893530e64ca. It has one inbound rule allowing all traffic from source sg-03df368a62ee282a4.
- EC2 > Instances**: Displays a list of EC2 instances. One instance, "dev-ec2-insta..." (ID: i-02a7bf505a0b5293), is selected. It is currently running and assigned to the "t3.micro" instance type. It is located in the "eu-north-1b" availability zone and has a public IPv4 address of 51.20.84.168.

← X ⓘ <https://100.31.242.176>

Terraform .

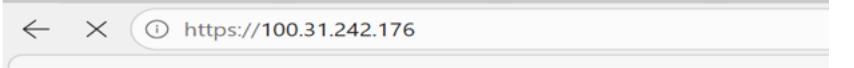
Question 3

```
@Zaheena1 [?] /workspaces/Lab_exam/ansible (main) $ cat <<EOF > hosts
> [ec2]
> 100.31.242.176
>
> [ec2:vars]
> ansible_user=ec2-user
> ansible_ssh_private_key_file=~/ssh/id_ed25519
> ansible_ssh_common_args=' -o StrictHostKeyChecking=no'
> EOF
@Zaheena1 [?] /workspaces/Lab_exam/ansible (main) $
```

```
@Zaheena1 ② /workspaces/Lab_exam/ansible (main) $ cat <<EOF > ansible.cfg
> [defaults]
> host_key_checking = False
> inventory = ./hosts
> interpreter_python = /usr/bin/python3
> EOF
@Zaheena1 ② /workspaces/Lab_exam/ansible (main) $
```

```
> ^C
@Zaheena1 ② /workspaces/Lab_exam/ansible (main) $ cat <<EOF > my-playbook.yml
> ---
> - name: Configure Web Server on Q2 Instance
>   hosts: ec2
>   become: true
>   tasks:
>     - name: Update all packages
>       dnf:
>         name: "*"
>         state: latest
>
>     - name: Stop and disable Nginx (from Q2)
>       service:
>         name: nginx
>         state: stopped
>         enabled: false
>       ignore_errors: true # Proceed even if nginx isn't there
>
>     - name: Install Apache HTTPD
>       dnf:
>         name: httpd
>         state: present
>
```

```
> ^C
@Zaheena1 ② /workspaces/Lab_exam/ansible (main) $ cat <<EOF > my-playbook.yml
> ---
> - name: Configure Web Server on Q2 Instance
>   hosts: ec2
>   become: true
>   tasks:
>     - name: Update all packages
>       dnf:
>         name: "*"
>         state: latest
>
>     - name: Stop and disable Nginx (from Q2)
>       service:
>         name: nginx
>         state: stopped
>         enabled: false
>       ignore_errors: true # Prevents failure if Nginx isn't installed
>
```



```
← × ⓘ https://100.31.242.176
```

It works!

Cleanup:

```
[root@ip-172-31-10-10 ~]# cd /workspaces/Lab_exam/lab_q2
[root@ip-172-31-10-10 lab_q2]# terraform destroy --auto-approve
data.http.myip: Reading...
aws_key_pair.ssh_key: Refreshing state... [id=serverkey]
aws_vpc.myapp_vpc: Refreshing state... [id=vpc-008116b4d00179796]
aws_subnet.myapp_subnet_1: Refreshing state... [id=subnet-0d140b7671c7aac8b]
aws_internet_gateway.myapp_igw: Refreshing state... [id=igw-0a7ca1fd1a9db5a6a]
aws_default_security_group.default_sg: Refreshing state... [id=sg-00a11a7d95856b56f]
aws_default_route_table.main_rt: Refreshing state... [id=rtb-09b7e1fa12c7b506a]
aws_instance.myapp_server: Refreshing state... [id=i-0c0fb8c02c802e47f]

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

Terraform will perform the following actions:

# aws_default_route_table.main_rt will be destroyed
- resource "aws_default_route_table" "main_rt" {
    - arn          = "arn:aws:ec2:us-east-1:819244121124:route-table/rtb-09b7e1fa12c7b506a" -> null
    - default_route_table_id = "rtb-09b7e1fa12c7b506a" -> null
    - id           = "rtb-09b7e1fa12c7b506a" -> null
    - owner_id     = "819244121124" -> null
    - propagating_vgws = [] -> null
    - route        = [
        - {
```