

Name: Sadaf Riaz

Roll no: 2023-BSE-077

Class: BSE5-B

Repo link: https://github.com/SadafRiaz077/Lab_exam

Lab Exam

```
▶ @SadafRiaz077 →/workspaces/Lab_exam (main) $ aws iam create-group --group-name SoftwareEngineering
{
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPAV4QTXR5S2GC7SEZY0",
        "Arn": "arn:aws:iam::404842057573:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:35:36+00:00"
    }
}
@SadafRiaz077 →/workspaces/Lab_exam (main) $ 
```

```
▶ @SadafRiaz077 →/workspaces/Lab_exam (main) $ aws iam get-group --group-name SoftwareEngineering
{
    "Users": [],
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPAV4QTXR5S2GC7SEZY0",
        "Arn": "arn:aws:iam::404842057573:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:35:36+00:00"
    }
}
@SadafRiaz077 →/workspaces/Lab_exam (main) $ 
```

```
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ aws iam create-user --user-name Sadaf
{
    "User": {
        "Path": "/",
        "UserName": "Sadaf",
        "UserId": "AIDAV4QTXR5SRNQRWQZFO",
        "Arn": "arn:aws:iam::404842057573:user/Sadaf",
        "CreateDate": "2026-01-19T07:36:51+00:00"
    }
}
@SadafRiaz077 →/workspaces/Lab_exam (main) $ 
```

```
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ aws iam get-user --user-name Sadaf
{
    "User": {
        "Path": "/",
        "UserName": "Sadaf",
        "UserId": "AIDAV4QTXR5SRNQRWQZFO",
        "Arn": "arn:aws:iam::404842057573:user/Sadaf",
        "CreateDate": "2026-01-19T07:36:51+00:00"
    }
}
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ 
```

```
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ aws iam add-user-to-group \
--user-name Sadaf \
--group-name SoftwareEngineering
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ 
```

```
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ aws iam get-group --group-name SoftwareEngineering
{
    "Users": [
        {
            "Path": "/",
            "UserName": "Sadaf",
            "UserId": "AIDAV4QTXR5SRNQRWQZFO",
            "Arn": "arn:aws:iam::404842057573:user/Sadaf",
            "CreateDate": "2026-01-19T07:36:51+00:00"
        }
    ],
    "Group": {
        "Path": "/",
        "GroupName": "SoftwareEngineering",
        "GroupId": "AGPAV4QTXR5S2GC7SEZY0",
        "Arn": "arn:aws:iam::404842057573:group/SoftwareEngineering",
        "CreateDate": "2026-01-19T07:35:36+00:00"
    }
}
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ 
```

```
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ aws iam list-policies --scope AWS --query "Policies[?PolicyName=='AdministratorAccess']"
[
    {
        "PolicyName": "AdministratorAccess",
        "PolicyId": "ANPAIWBCKSKIEE64ZLYK",
        "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 3,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2015-02-06T18:39:46+00:00",
        "UpdateDate": "2015-02-06T18:39:46+00:00"
    }
]
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ 
```

```

● @SadafRiaz077 →/workspaces/Lab_exam (main) $ aws iam attach-group-policy \
--group-name SoftwareEngineering \
--policy-arn arn:aws:iam::aws:policy/AdministratorAccess
○ @SadafRiaz077 →/workspaces/Lab_exam (main) $ 

```

```

● @SadafRiaz077 →/workspaces/Lab_exam (main) $ aws iam list-attached-group-policies \
--group-name SoftwareEngineering
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        }
    ]
}
○ @SadafRiaz077 →/workspaces/Lab_exam (main) $ 

```

AWS

Users (4) <small>Info</small>											
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.											
<input type="button" value="Search"/> <input type="button" value="Delete"/> <input type="button" value="Create user"/>											
<input type="button" value="Search"/>											
User name	Path	Group:	Last activity	MFA	Password age	Console last sign-in	Access key ID	Active key age	Active key age	Access	Access
key	/	0	4 minutes ago	-	-	-	Active - AKIAV4QTXRS...	8 minutes	8 minutes	4	4
kdg	/	0	2 days ago	-	4 days	-	Active - AKIAV4QTXRS...	4 days	4 days	2	2
project	/	0	Yesterday	-	2 days	-	Active - AKIAV4QTXRS...	2 days	2 days	Yesterda	Yesterda
Sadaf	/	1	-	-	-	-	-	-	-	-	-

IAM > Users

Identity and Access Management (IAM)

Dashboard

▼ Access Management

- User groups
- Users**
- Roles
- Policies
- Identity providers
- Account settings
- Root access management
- Temporary delegation requests

Users (4) Info

An IAM user is an identity with long-term cred

User name
key
kdg
project
Sadaf

The screenshot shows the AWS IAM User groups page. At the top, there is a search bar and a navigation bar with the user name "Sadaf Riaz (4048-4205-7573)" and "Sadaf Riaz". Below the search bar, there is a table titled "User groups (1) Info". The table has columns for "Group name" (SoftwareEngineering), "Users" (empty), "Permissions" (Defined), and "Creation time" (8 minutes ago). There are buttons for "Delete" and "Create group".

The screenshot shows the AWS IAM SoftwareEngineering group details page. The left sidebar shows the navigation path: IAM > User groups > SoftwareEngineering. The main content area has tabs for "Summary", "Users (1)", "Permissions", and "Access Advisor". The "Summary" tab shows the user group name "SoftwareEngineering" and creation time "January 18, 2026, 23:35 (UTC-08:00)". The "Permissions" tab shows a table for "Permissions policies (1) Info". It lists one policy: "AdministratorAccess" (AWS managed - job function). There are buttons for "Edit", "Delete", "Simulate", "Remove", and "Add permissions".

This screenshot is identical to the one above, showing the AWS IAM SoftwareEngineering group details page. The left sidebar shows the navigation path: IAM > User groups > SoftwareEngineering. The main content area has tabs for "Summary", "Users (1)", "Permissions", and "Access Advisor". The "Summary" tab shows the user group name "SoftwareEngineering" and creation time "January 18, 2026, 23:35 (UTC-08:00)". The "Permissions" tab shows a table for "Permissions policies (1) Info". It lists one policy: "AdministratorAccess" (AWS managed - job function). There are buttons for "Edit", "Delete", "Simulate", "Remove", and "Add permissions".

Q 2

```
● @SadafRiaz077 →/workspaces/Lab_exam (main) $ mkdir q2-terraform
cd q2-terraform

touch main.tf
touch variables.tf
touch terraform.tfvars
touch outputs.tf
touch entry-script.sh
○ @SadafRiaz077 →/workspaces/Lab_exam/q2-terraform (main) $ 
```

Variables.tf

```
● @SadafRiaz077 →/workspaces/Lab_exam/q2-terraform (main) $ nano variables.tf
● @SadafRiaz077 →/workspaces/Lab_exam/q2-terraform (main) $ cat variables.tf
variable "vpc_cidr_block" {
  type = string
}

variable "subnet_cidr_block" {
  type = string
}

variable "availability_zone" {
  type = string
}

variable "env_prefix" {
  type = string
}

variable "instance_type" {
  type = string
}

○ @SadafRiaz077 →/workspaces/Lab_exam/q2-terraform (main) $ 
```

Main.tf

Ec2, route table, subnet

```

@SadafRiaz077 ~/workspaces/Lab_Exam/q2-terraform (main) $ nano main.tf

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"
  }
}

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"
  }
}

```

```

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" "serverkey" {
  key_name = "serverkey"
}

```

Locals

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

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

Security group

```
resource "aws_instance" "myapp_ec2" {
  ami                      = data.aws_ami.amazon_linux2.id
  instance_type             = var.instance_type
  subnet_id                 = aws_subnet.myapp_subnet.id
  availability_zone         = var.availability_zone
  vpc_security_group_ids   = [aws_default_security_group.default_sg.id]
  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"
  }
}
```

init

```
@SadafRiaz077 →/workspaces/Lab_exam/q2-terraform (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/http from the dependency lock file
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/http v3.5.0
- Using previously-installed hashicorp/aws v6.28.0
```

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 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.

Apply

```
• @SadafRiaz077 →/workspaces/Lab_exam/q2-terraform (main) $ terraform apply -auto-approve
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 1s [id=https://icanhazip.com]
aws_key_pair.serverkey: Refreshing state... [id=serverkey]
data.aws_ami.amazon_linux2: Reading...
aws_vpc.myapp_vpc: Refreshing state... [id=vpc-0b99b091aee313055]
data.aws_ami.amazon_linux2: Read complete after 1s [id=ami-057e2bc910cc38f26]
aws_subnet.myapp_subnet: Refreshing state... [id=subnet-0f5490d28c070f027]
aws_internet_gateway.myapp_igw: Refreshing state... [id=igw-0bd54eae6a10dc2bb]
aws_default_security_group.default_sg: Refreshing state... [id=sg-082c25e7aec0723fb]
aws_default_route_table.myapp_rt: Refreshing state... [id=rtb-04a70a64e7063bcff]
aws_instance.myapp_ec2: Refreshing state... [id=i-0a4315feba50cfa21]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

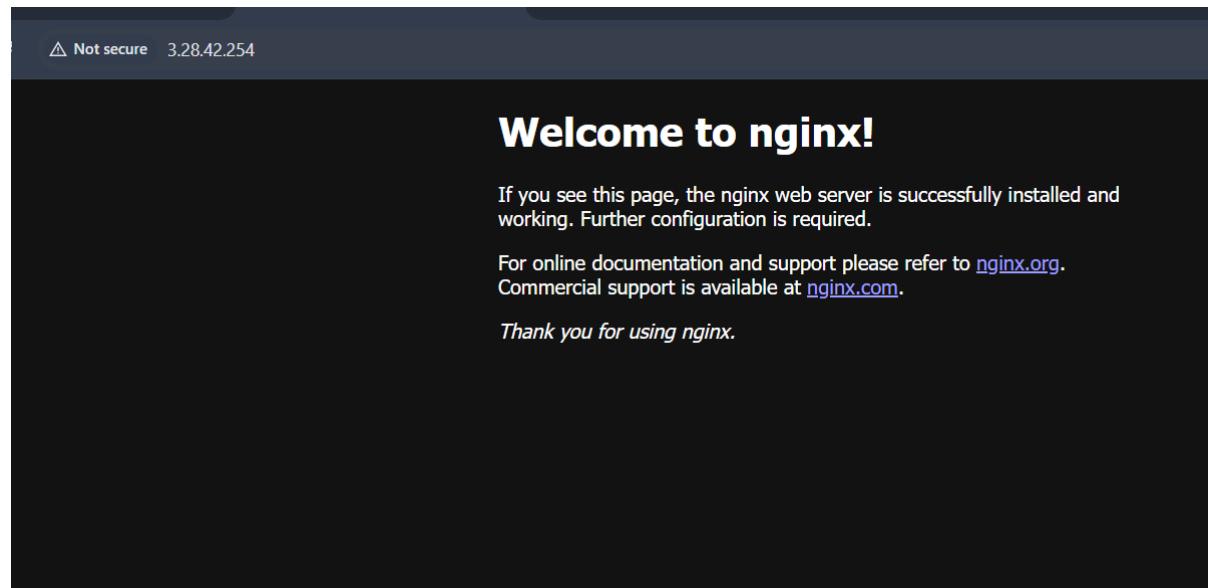
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.

Outputs:
```

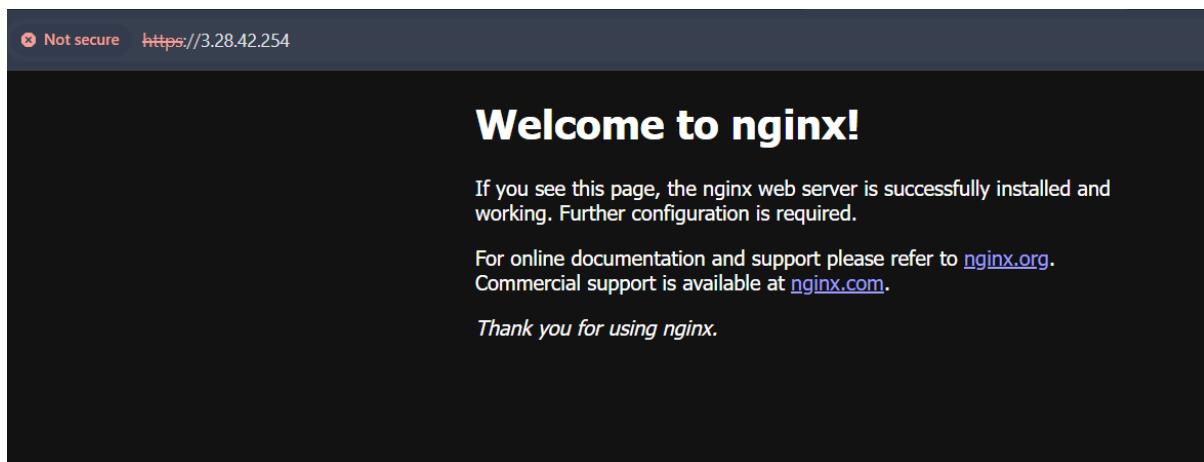
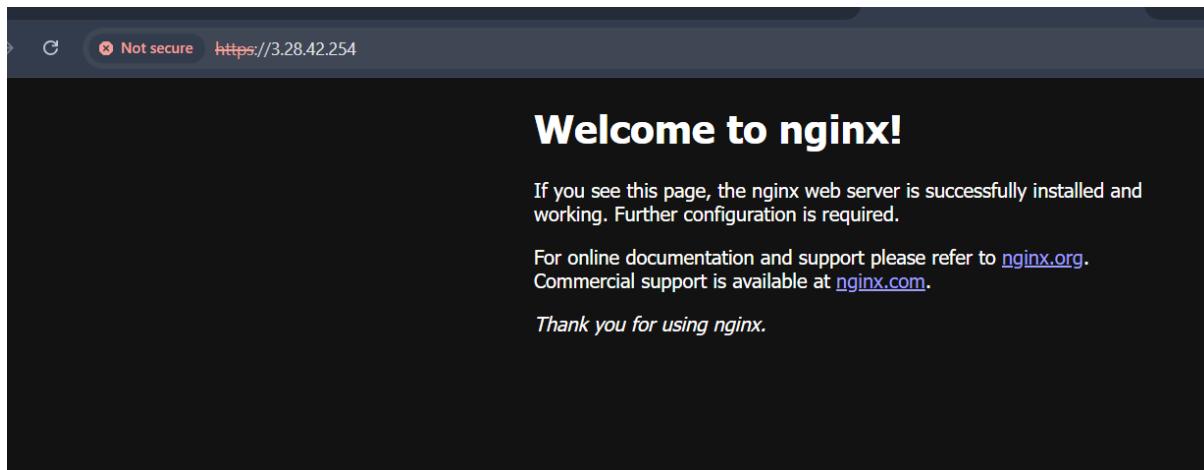
```
ec2_public_ip = "3.28.42.254"
@SadafRiaz077 →/workspaces/Lab_exam/q2-terraform (main) $ ssh -i ~/.ssh/id_ed25519 ec2-user@3.28.42.254
```

ssh

```
@SadafRiaz077 →/workspaces/Lab_exam/q2-terraform (main) $ ssh -i ~/.ssh/id_ed25519 ec2-user@3.28.42.254
Last login: Mon Jan 19 08:24:54 2026 from 20.192.21.54
      , #_
~\_\ ##### Amazon Linux 2
~~ \_\#####\
~~ \###| AL2 End of Life is 2026-06-30.
~~ \#/ __-
~~ V`' '-->
~~ / A newer version of Amazon Linux is available!
~~ ._. / Amazon Linux 2023, GA and supported until 2028-03-15.
/m/ / https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-10-0-10-171 ~]$
```



Convert to https



Question3

```
● @SadafRiaz077 ~ /Lab_Exam/ansible/ansible $ nano hosts
[ec2]
3.28.42.254

[ec2:vars]
ansible_user=ec2-user
ansible_ssh_private_key_file=~/ssh/id_ed25519
ansible_ssh_common_args=' -o StrictHostKeyChecking=no'
● @SadafRiaz077 ~ /Lab_Exam/ansible/ansible $ [
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
GNU nano 7.2
[defaults]
inventory = ./hosts
host_key_checking = False
remote_python_interpreter = /usr/bin/python3
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
GNU nano 7.2
my-playbook.yml *
- name: Get public IPv4
  uri:
    url: http://169.254.169.254/latest/meta-data/public-ipv4
    headers:
      X-aws-ec2-metadata-token: "{{ imds_token.content }}"
  register: ec2_public_ip

- name: Get public hostname
  uri:
    url: http://169.254.169.254/latest/meta-data/public-hostname
    headers:
      X-aws-ec2-metadata-token: "{{ imds_token.content }}"
  register: ec2_public_hostname

- name: Print EC2 public IP
  debug:
    msg: "EC2 Public IP: {{ ec2_public_ip.content }}"

- name: Restart httpd service
  service:
    name: httpd
    state: restarted
```

```
TASK [Print EC2 public IP] ****
ok: [3.28.42.254] => {
  "msg": "EC2 public IP is "
}

TASK [Print EC2 public hostname] ****
ok: [3.28.42.254] => {
  "msg": "EC2 public hostname is "
}

TASK [Restart httpd] ****
changed: [3.28.42.254]

PLAY RECAP ****
3.28.42.254 : ok=13  changed=1  unreachable=0  failed=0  skipped=0  rescued=0  ignored=4

@SadafRiaz077 ~~/Lab_exam/ansible/ansible $ []
```

```
sed -i '$([ ! ${EC2_PUBLIC_IP} ] && echo "host_name: ${EC2_PUBLIC_IP}" )' my-playbook.yml
@sadafriaz@77 ~~/Lab_exam/ansible/ansible $ ansible-playbook -i hosts my-playbook.yml

PLAY [Configure EC2 Web Server] *****

TASK [Gathering Facts] *****
[WARNING]: Platform linux on host 3.28.42.254 is using the discovered Python interpreter at /usr/bin/python3.7, but future installation of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.16/reference_appendices/interpreter_discovery.html for more information.
ok: [3.28.42.254]

TASK [Update system packages] *****
ok: [3.28.42.254]

TASK [Stop and disable Nginx if installed] *****
fatal: [3.28.42.254]: FAILED! => {"changed": false, "msg": "Could not find the requested service nginx: host"}
...ignoring

TASK [Remove Nginx package if installed] *****
ok: [3.28.42.254]

TASK [Install httpd (Apache)] *****
ok: [3.28.42.254]

TASK [Start and enable httpd service] *****
ok: [3.28.42.254]

TASK [Get IMDSv2 token] *****
ok: [3.28.42.254]

TASK [Get public IPv4] *****
fatal: [3.28.42.254]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'dict object' has no attribute 'content'. 'dict object' has no attribute 'content'\n\nThe error appears to be in '/home/codespace/Lab_exam/ansible/my-playbook.yml': line 52, column 7, but may\nbe elsewhere in the file depending on the exact syntax problem.\n\nThe offending line appears to be:\n\n      - name: Get public IPv4\n          ^ here\n\"}

PLAY RECAP *****
3.28.42.254 : ok=7    changed=0   unreachable=0   failed=1   skipped=0   rescued=0   ignored=1
```