

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 SoftwareEngineering
{
  "Group": {
    "Path": "/",
    "GroupName": "SoftwareEngineering",
    "GroupId": "AGPAUHX42RXHCKQUE7IL",
    "Arn": "arn:aws:iam::291506017390:group/SoftwareEngineering",
    "CreateDate": "2026-01-19T08:13:35+00:00"
  }
}
```

q1_group_details.png

```
@mariamalik11 /workspaces/lab_exam (main) $ aws iam get-group --group-name SoftwareEngineering
{
  "Users": [],
  "Group": {
    "Path": "/",
    "GroupName": "SoftwareEngineering",
    "GroupId": "AGPAUHX42RXHCKQUE7IL",
    "Arn": "arn:aws:iam::291506017390:group/SoftwareEngineering",
    "CreateDate": "2026-01-19T08:13:35+00:00"
  }
}
```

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": "AIDAUXHY42RXAX5HFIMVN",
    "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": "ANPAIWMBCSKIEE64ZLYK",
    "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) $
```

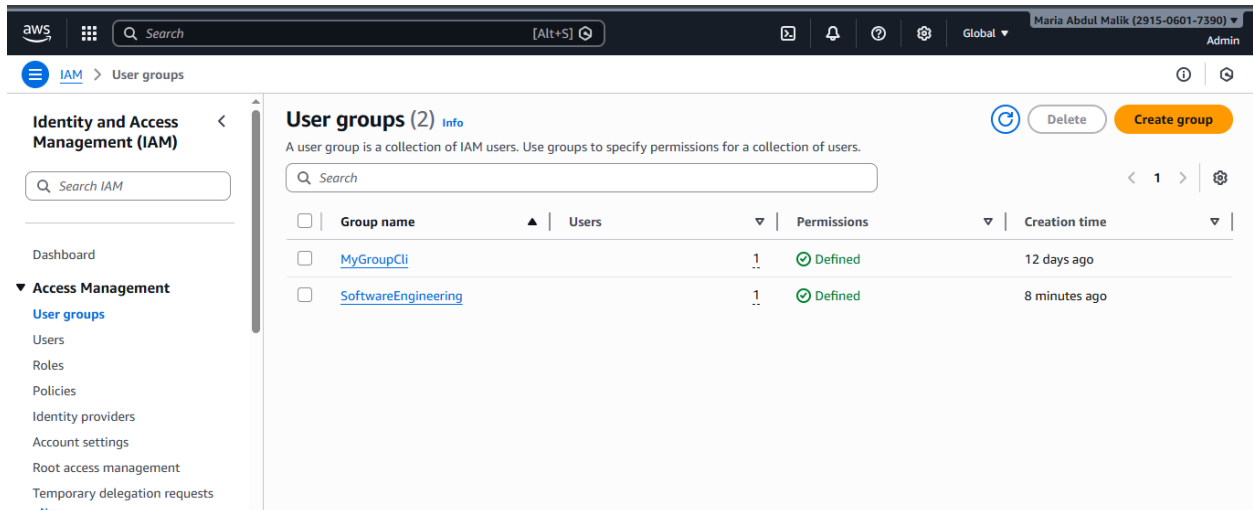
q1_attach_admin_policy.png

```
@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) $
```

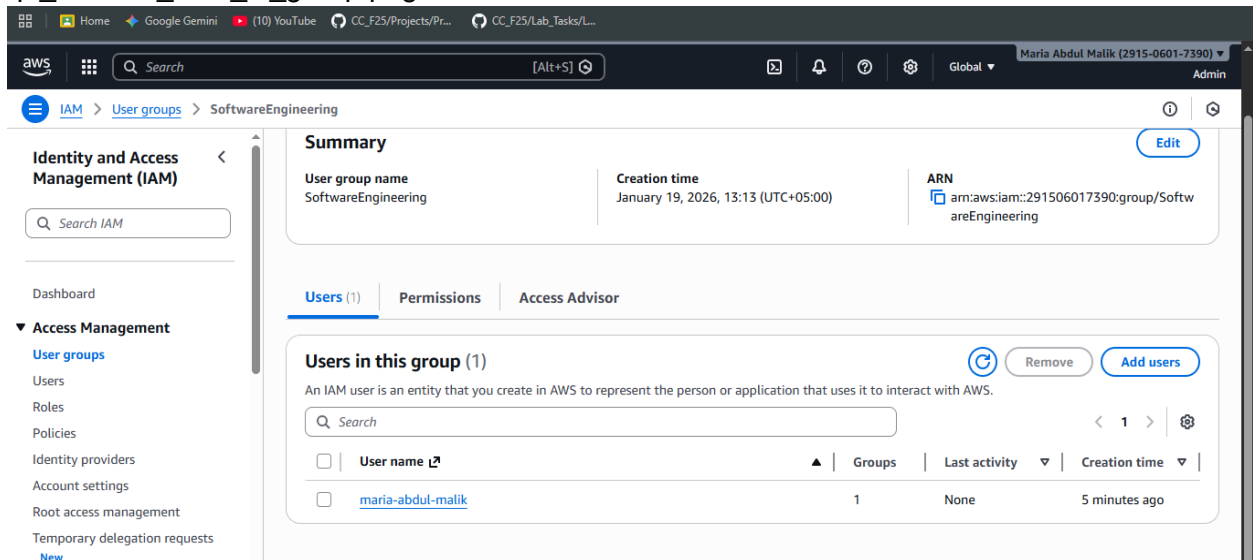
q1_list_group_policies.png

```
@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_exam (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/nginx/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"
+     },
+   ],
+   ingress              = []
+   name                 = (known after apply)
+   owner_id             = (known after apply)
+   vpc_id              = (known after apply)
}
```



```

+ 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-0819532e60f0c0e0e]

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

The screenshot shows the AWS Management Console interface for the 'VPC' service. The main content area displays 'Your VPCs (3)' with a table listing the following VPCs:

Name	VPC ID	State	Encryption c...	Encryption control ...
frontend-backend-lab-vpc	vpc-0bdc17c3e3a2285b6	Available	-	-
dev-vpc	vpc-0f62335e635c24298	Available	-	-
-	vpc-0e3a7ba238e097490	Available	-	-

The left sidebar shows the navigation menu with 'Virtual private cloud' expanded, listing options like 'Your VPCs', 'Subnets', 'Route tables', 'Internet gateways', 'Egress-only internet gateways', 'DHCP option sets', 'Elastic IPs', 'Managed prefix lists', 'Endpoints', and 'Endpoint services'.

Q2_console_subnet.png

aws [Search] [Alt+S] Middle East (UAE) Lab8User

VPC > Subnets > subnet-0de6569a2f0e3b744

subnet-0de6569a2f0e3b744 / dev-subnet-1

Details

Subnet ID subnet-0de6569a2f0e3b744	Subnet ARN arn:aws:ec2:me-central-1:291506017390:subnet/subnet-0de6569a2f0e3b744	State Available	Block Public Access Off
IPv4 CIDR 10.0.10.0/24	Available IPv4 addresses 250	IPv6 CIDR -	IPv6 CIDR association ID -
Availability Zone mec1-az1 (me-central-1a)	VPC vpc-0f62335e635c24298 dev-vpc	Route table rtb-0a0f15e4b7777177c dev-rt	Network ACL acl-0836eb88efcd27a9a
Default subnet No	Auto-assign public IPv4 address No	Auto-assign IPv6 address No	Outpost ID -
IPv4 CIDR reservations -	IPv6 CIDR reservations -	IPv6-only No	Hostname type IP name
Resource name DNS A record Disabled	Resource name DNS AAAA record Disabled	DNS64 Disabled	Owner 291506017390

q2_console_igw.png

aws [Search] [Alt+S] Middle East (UAE) Maria Abdul Malik (2915-0601-7390) Lab8User

VPC > Internet gateways > igw-0d683ea9737790f19

igw-0d683ea9737790f19 / dev-igw

Details

Internet gateway ID igw-0d683ea9737790f19	State Attached	VPC ID vpc-0f62335e635c24298 dev-vpc	Owner 291506017390
---	--------------------------	--	------------------------------

Tags (1)

Key	Value
Name	dev-igw

q2_console_route_table.png

aws [Search] [Alt+S] Middle East (UAE) Maria Abdul Malik (2915-0601-7390) Lab8User

VPC > Route tables > rtb-0a0f15e4b7777177c

rtb-0a0f15e4b7777177c / dev-rt

Details

Route table ID rtb-0a0f15e4b7777177c	Main Yes	Explicit subnet associations -	Edge associations -
VPC vpc-0f62335e635c24298 dev-vpc	Owner ID 291506017390		

Routes | Subnet associations | Edge associations | Route propagation | Tags

Routes (2)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-0d683ea9737790f19	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table

q2_console_sg.png

The screenshot shows the AWS Management Console interface for a Security Group. The breadcrumb navigation is VPC > Security Groups > sg-0cb81405edf7d7b89 - default. The left sidebar lists various AWS services, with 'Security' and 'Security groups' highlighted. The main content area displays the details for the Security Group 'sg-0cb81405edf7d7b89 - default'. The details include: Security group name (default), Security group ID (sg-0cb81405edf7d7b89), Description (default VPC security group), VPC ID (vpc-0f62335e635c24298), Owner (291506017390), Inbound rules count (3 Permission entries), and Outbound rules count (1 Permission entry). Below the details, there are tabs for Inbound rules, Outbound rules, Sharing, VPC associations, and Tags. The 'Inbound rules' tab is selected, showing a table with 3 rules. The table has columns: Name, Security group rule ID, IP version, Type, Protocol, and Port. The rules are: 1. Name: -, Security group rule ID: sgr-084476200a6109994, IP version: IPv4, Type: SSH, Protocol: TCP, Port: 22. 2. Name: -, Security group rule ID: sgr-05bcc7ad16921e1fc, IP version: IPv4, Type: HTTPS, Protocol: TCP, Port: 443. There are also buttons for 'Manage tags' and 'Edit inbound rules'.

Name	Security group rule ID	IP version	Type	Protocol	Port
-	sgr-084476200a6109994	IPv4	SSH	TCP	22
-	sgr-05bcc7ad16921e1fc	IPv4	HTTPS	TCP	443

q2_console_ec2.png

The screenshot shows the AWS Management Console interface for an EC2 instance. The breadcrumb navigation is EC2 > Instances > i-0fdbb82cc7e99be09. The left sidebar lists various AWS services, with 'EC2' and 'Instances' highlighted. The main content area displays the details for the EC2 instance 'i-0fdbb82cc7e99be09 (MyServer)'. The details include: Instance ID (i-0fdbb82cc7e99be09), Public IPv4 address (51.112.142.8), Private IPv4 addresses (172.31.24.116), Instance state (Running), Public DNS (ec2-51-112-142-8.me-central-1.compute.amazonaws.com), Private IP DNS name (ip-172-31-24-116.me-central-1.compute.internal), Instance type (t3.micro), VPC ID (vpc-0e3a7ba238e097490), and AWS Compute Optimizer finding (Opt-in to AWS Compute Optimizer for recommendations). There are also buttons for 'Connect', 'Instance state', and 'Actions'.

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0fdbb82cc7e99be09	51.112.142.8	172.31.24.116

q2_https_browser.png

The screenshot shows a web browser window with the URL <https://51.112.42.127/> in the address bar. The page content displays the text "This is Maria's Terraform environment." in a large, bold, black font.