

Task 10

Terraform module:

A **Terraform module** is a reusable block of Terraform code that groups related resources into a single logical unit. Instead of writing the same configuration repeatedly, you place it inside a module and simply call it whenever needed. This helps keep your infrastructure clean, organized, and scalable. Modules improve reusability, reduce duplication, and allow teams to standardize deployments across different environments such as dev, test, and production. In simple terms, a module acts like a template that lets you deploy the same set of resources multiple times with different inputs.

Screenshots of Terraform code:

Ec2:

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER:** Shows a file tree with a folder named "MODULES" containing ".terraform", "ec", and "main.tf". Inside "ec", there are files ".terraform.lock.hcl", "main.tf", "provider.tf", "terraform.state", and "terraform.state.backup".
- CODE EDITOR:** Displays the contents of "main.tf" which defines an AWS instance named "web-server".

```
resource "aws_instance" "web-server" {
  ami           = "ami-093a7f5fbac13ff67"
  instance_type = "t3.micro"
  tags = [
    { Name = "web-server" }
  ]
}
```
- TERMINAL:** Shows the command \$ terraform destroy being run in a terminal window. The output indicates that Terraform will destroy all managed infrastructure and asks for confirmation with "Enter a value: yes". It then lists several module instances and concludes with "Destroy complete! Resources: 1 destroyed".

```
SRINATH@leo MINGW64 ~/OneDrive/Desktop/AWS & DEVOPS/Devops/Modules
$ terraform destroy
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

module.ec2.aws_instance.web-server: Destroying... [id=i-095b804fa574d2ea4]
module.ec2.aws_instance.web-server: Still destroying... [id=i-095b804fa574d2ea4, 0m10s elapsed]
module.ec2.aws_instance.web-server: Still destroying... [id=i-095b804fa574d2ea4, 0m20s elapsed]
module.ec2.aws_instance.web-server: Still destroying... [id=i-095b804fa574d2ea4, 0m30s elapsed]
module.ec2.aws_instance.web-server: Still destroying... [id=i-095b804fa574d2ea4, 0m40s elapsed]
module.ec2.aws_instance.web-server: Destruction complete after 40s

Destroy complete! Resources: 1 destroyed.
```

S3:

The screenshot shows the Visual Studio Code interface with the Terraform extension installed. The left sidebar displays a file tree with modules for .terraform, ec, S3, vpc, and provider. The main editor pane contains the following Terraform code:

```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "mybucket148293801"
  acl   = "private"
  tags = {
    Name     = "MyBucket"
    Environment = "Dev"
  }
}
```

The bottom status bar indicates the code is at Line 11, Column 1, with 4 spaces, using UTF-8 encoding, and is part of the Terraform workspace. A tooltip in the Problems panel states: "acl is deprecated: Reason: ** Terraform [Ln 5, Col 3]."

Vpc:

The screenshot shows the Visual Studio Code interface with the Terraform extension installed. The left sidebar displays a file tree with modules for .terraform, ec, S3, vpc, and provider. The main editor pane contains the following Terraform code:

```
resource "aws_internet_gateway" "igw" {
}

resource "aws_default_route_table" "public_rt" {
  default_route_table_id = aws_vpc.main.default_route_table.id
}

route {
  cidr_block = "0.0.0.0/0"
  gateway_id = aws_internet_gateway.igw.id
}

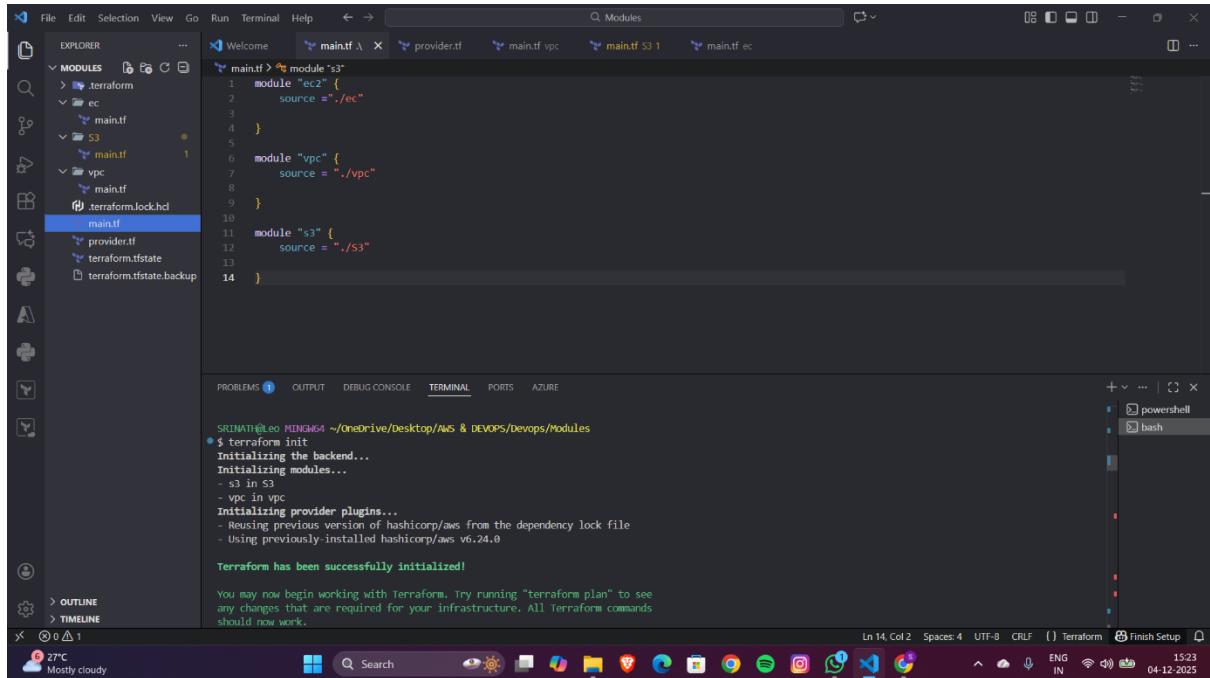
tags = {
  Name = "public-route-table"
}

resource "aws_route_table_association" "public_1_assoc" {
  subnet_id   = aws_subnet.public_1.id
  route_table_id = aws_default_route_table.public_rt.id
}
```

The bottom status bar indicates the code is at Line 46, Column 1, with 4 spaces, using UTF-8 encoding, and is part of the Terraform workspace. A tooltip in the Problems panel states: "acl is deprecated: Reason: ** Terraform [Ln 5, Col 3]."

Commands:

Terraform init:



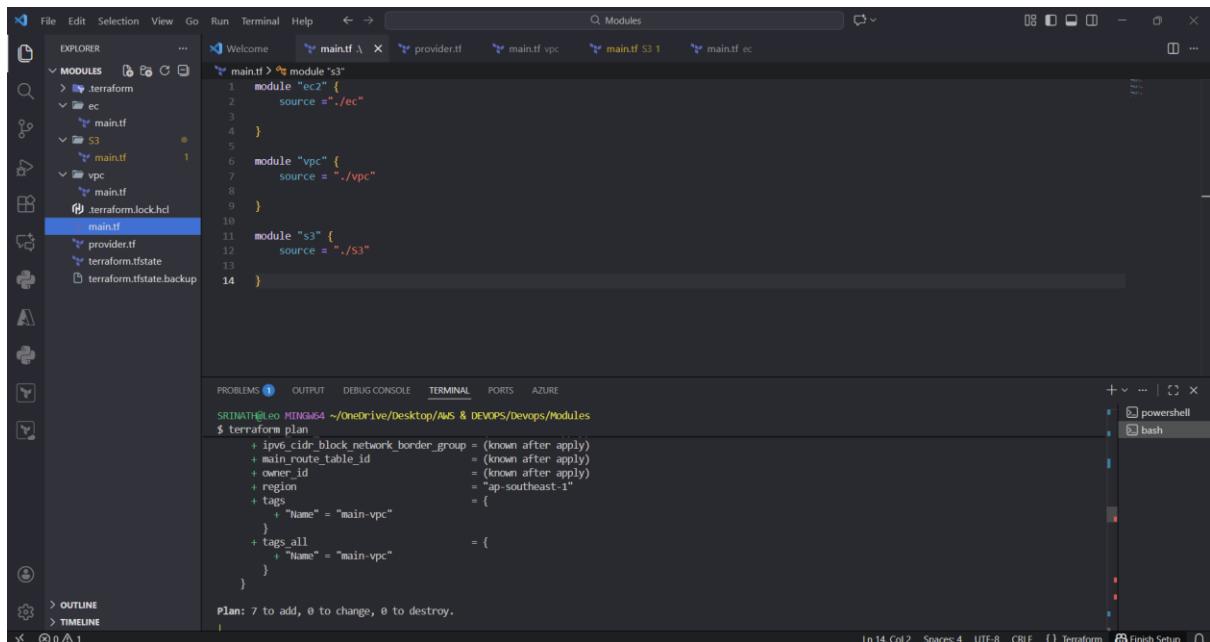
The screenshot shows the VS Code interface with the Terraform extension installed. The Explorer sidebar shows a directory structure with modules for EC, S3, and VPC. The main editor tab displays a Terraform configuration file (main.tf) containing three module blocks: ec, vpc, and s3. Below the code editor is a terminal window showing the output of the 'terraform init' command. The terminal output indicates that Terraform is initializing the backend, modules, and provider plugins, and successfully initializes the configuration.

```
SRINATH@leo MINGW64 ~/OneDrive/Desktop/AWS & DEVOPS/Devops/Modules
$ terraform init
Initializing the backend...
Initializing modules...
- s3 in S3
- vpc in vpc
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.24.0

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.
```

Terraform plan:



The screenshot shows the VS Code interface with the Terraform extension installed. The Explorer sidebar shows a directory structure with modules for EC, S3, and VPC. The main editor tab displays a Terraform configuration file (main.tf) containing three module blocks: ec, vpc, and s3. Below the code editor is a terminal window showing the output of the 'terraform plan' command. The terminal output shows the plan for creating a new VPC resource, including its IP range, route table, owner ID, region, and tags.

```
SRINATH@leo MINGW64 ~/OneDrive/Desktop/AWS & DEVOPS/Devops/Modules
$ terraform plan
+ ipv6.cidr_block_network_border_group = (known after apply)
+ main_route_table_id = (known after apply)
+ owner_id = (known after apply)
+ region = "ap-southeast-1"
+ tags =
  + "Name" = "main-vpc"
}
+ tags_all =
  + "Name" = "main-vpc"
}
```

Terraform apply:

The screenshot shows the Visual Studio Code interface with the Terraform extension installed. The left sidebar displays the project structure with files like `.terraform.lock.hcl`, `main.tf`, `provider.tf`, and `terraform.state.backup`. The main editor pane shows the `main.tf` file containing Terraform configuration for EC2, VPC, and S3 modules. The bottom terminal pane shows the command `terraform apply` being run, followed by confirmation steps and the final output: `Apply complete! Resources: 2 added, 0 changed, 0 destroyed.`

```
main.tf
1 module "ec2" {
2     source = "./ec"
3
4 }
5
6 module "vpc" {
7     source = "./vpc"
8
9 }
10
11 module "s3" {
12     source = "./s3"
13
14 }
```

```
SRINATH@leo MING64 ~/OneDrive/Desktop/AWS & DEVOPS/Devops/Modules
$ terraform 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

module.vpc.aws_subnet.public_1: Creating...
module.vpc.aws_subnet.public_3: Still creating... [00m10s elapsed]
module.vpc.aws_subnet.public_1: Creation complete after 1s [id=subnet-09042f4d5194194be]
module.vpc.aws_route_table_association.public_1_assoc: Creating...
module.vpc.aws_route_table_association.public_1_assoc: Creation complete after 1s [id=rtbassoc-01a1cb05fc16a7937]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

OUTPUT:

Ec2:

The screenshot shows the AWS Management Console EC2 Instances page. The left navigation menu is expanded to show options like Dashboard, AWS Global View, Instances, Images, and Elastic Block Store. The main content area displays a table of instances. One instance, named `web-server` with Instance ID `i-095b804fa574d2ea4`, is listed as `Running`. A modal window titled `Select an instance` is open, listing the same instance for selection. The top of the screen shows the browser URL as `ap-southeast-1.console.aws.amazon.com/ec2/home?region=ap-southeast-1#instances?v=3;case=true%5C;client=false;$regex=tags:false%5C;client:false`.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public
web-server	i-095b804fa574d2ea4	Running	t3.micro	Initializing		ap-southeast-1c	ec2-13-212-125-181.ap...	13.212

S3:

The screenshot shows the AWS S3 console interface. On the left, there's a navigation sidebar with sections like Buckets, Access management and security, Storage management and insights, and Account and organization settings. The main area displays a table of General purpose buckets. The table has columns for Name, AWS Region, and Creation date. Three buckets are listed:

Name	AWS Region	Creation date
mybucket148293801	Asia Pacific (Singapore) ap-southeast-1	December 4, 2025, 15:19:03 (UTC+05:30)
sinath-terraformstatefile	US West (N. California) us-west-1	December 1, 2025, 19:47:17 (UTC+05:30)
state123098	US West (N. California) us-west-1	December 5, 2025, 22:55:04 (UTC+05:30)

On the right, there are two cards: 'Account snapshot' and 'External access summary - new'. The bottom of the screen includes standard AWS navigation links like CloudShell, Feedback, and Console Mobile App.

VPC:

The screenshot shows the AWS VPC console interface. The left sidebar includes sections for 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 servers), Security (Network ACLs, Security groups), and PrivateLink and. The main area features a prominent blue callout box about VPC encryption control. Below it, the 'Your VPCs' section displays a table of existing VPCs. The table has columns for Name, VPC ID, State, Encryption controls, Block Public, IPv4 CIDR, and Actions. Two VPCs are listed:

Name	VPC ID	State	Encryption controls	Block Public	IPv4 CIDR
main-vpc	vpc-0291cdd75bfeaa28c	Available	-	Off	10.0.0.0/16
-	vpc-0518cc416f950272e	Available	-	Off	172.31.0.0/16

At the bottom, there's a note to 'Select a VPC above'. The bottom of the screen includes standard AWS navigation links like CloudShell, Feedback, and Console Mobile App.

Subnet:

The screenshot shows the AWS VPC Subnets page. The left sidebar includes sections for VPC dashboard, Virtual private cloud (Your VPCs, Subnets), Security (Network ACLs, Security groups), and Private link and CloudShell. The main content area displays a table titled "Subnets (4) Info" with columns: Name, Subnet ID, State, VPC, Block Public..., IPv4 CIDR, and IPv6 CIDR. The subnets listed are:

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR	IPv6 CIDR
PublicSubnet1	subnet-05042f4d5194194be	Available	vpc-0291cd75bfeea28c main...	Off	10.0.1.0/24	-
-	subnet-0e5c664dbd4a9d459	Available	vpc-0518cc416f950272e	Off	172.31.32.0/20	-
-	subnet-038428a6ccba79763	Available	vpc-0518cc416f950272e	Off	172.31.16.0/20	-
-	subnet-05c97d846dc00441	Available	vpc-0518cc416f950272e	Off	172.31.0.0/20	-

Route table:

The screenshot shows the AWS VPC Route tables page. The left sidebar includes sections for VPC dashboard, Virtual private cloud (Your VPCs, Subnets, Route tables), Security (Network ACLs, Security groups), and Private link and CloudShell. The main content area displays a table titled "Route tables (2) Info" with columns: Name, Route table ID, Explicit subnet associ..., Edge associations, Main, VPC, and Owner ID. The route tables listed are:

Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC	Owner ID
-	rtb-08aae774fd2acb93	-	-	Yes	vpc-0518cc416f950272e	405161739265
public-route-table	rtb-0f6f796fb01c7c799	subnet-09042f4d519419...	-	Yes	vpc-0291cd75bfeea28c main...	405161739265

Internet gateway:

The screenshot shows the AWS VPC dashboard with the 'Internet gateways' section selected. There are two internet gateways listed:

Name	Internet gateway ID	State	VPC ID	Owner
-	igw-067149c6c0b51848b	Attached	vpc-0518cc416f950272e	405161739265
main-igw	igw-0fd441861ab429fc	Attached	vpc-0291cdd75bfeea28c main-vpc	405161739265

A button labeled 'Create internet gateway' is visible at the top right.

Terraform destroy:

The screenshot shows the Visual Studio Code interface with the Terraform extension open. The code editor displays the main.tf file:

```
module "vpc" {
  source = "./vpc"
}

module "s3" {
  source = "./S3"
}
```

The terminal tab shows the output of a 'terraform destroy' command:

```
module.vpc.aws_default_route_table.public_rt: Destroying... [id=rtb-0f6f796fb01c7c799]
module.vpc.aws_default_route_table.public_rt: Destruction complete after 0s
module.vpc.aws_internet_gateway.igw: Destroying... [id=igw-0fd441861ab429fc]
module.s3.aws_s3_bucket.my_bucket: Destruction complete after 0s
module.vpc.aws_internet_gateway.igw: Destruction complete after 0s
module.vpc.aws_subnet.public_1: destruction complete after 0s
module.vpc.aws_vpc.main: Destroying... [id=vpc-0291cdd75bfeea28c]
module.vpc.aws_vpc.main: Destruction complete after 0s
module.ec2.aws_instance.web-server: Still destroying... [id=i-0ee0b963447faf76c, 0m10s elapsed]
module.ec2.aws_instance.web-server: Still destroying... [id=i-0ee0b963447faf76c, 0m20s elapsed]
module.ec2.aws_instance.web-server: Still destroying... [id=i-0ee0b963447faf76c, 0m30s elapsed]
module.ec2.aws_instance.web-server: Still destroying... [id=i-0ee0b963447faf76c, 0m40s elapsed]
module.ec2.aws_instance.web-server: Destruction complete after 40s
Destroy complete! Resources: 7 destroyed.
```