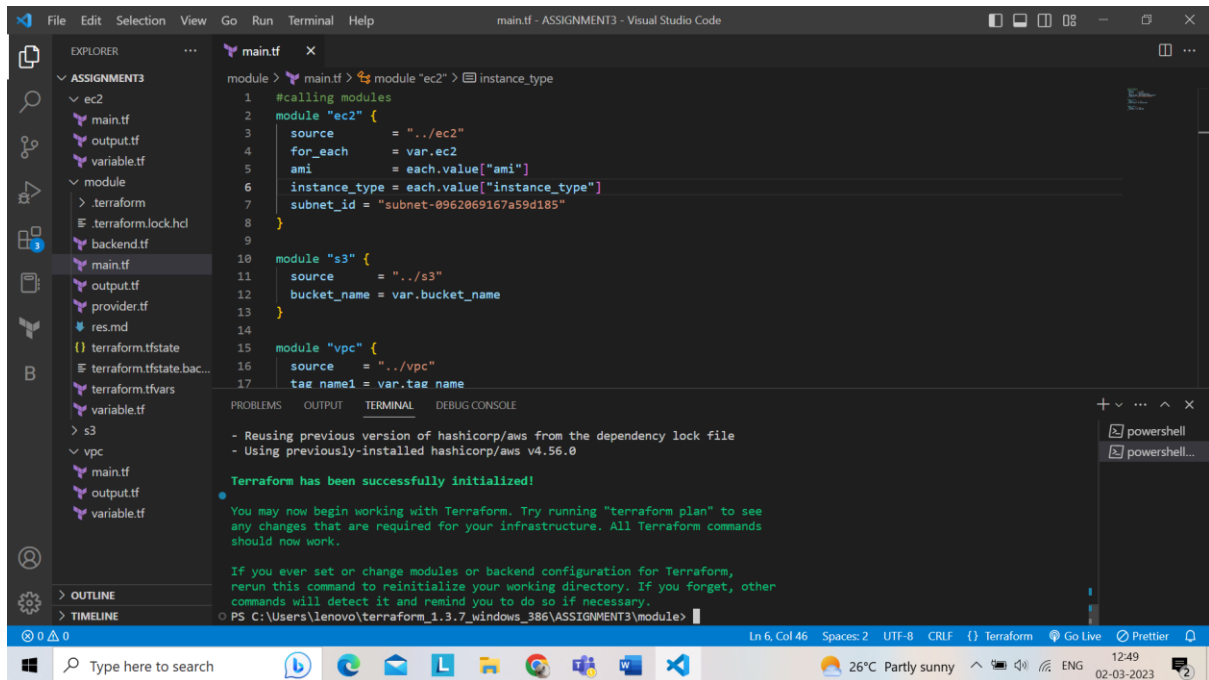


TERRAFORM INITIALIZED



The screenshot shows the Visual Studio Code interface with the 'main.tf' file open. The file contains Terraform configuration for an EC2 instance and an S3 bucket. The terminal window at the bottom displays the output of the 'terraform init' command, indicating that Terraform has been successfully initialized. The Explorer sidebar on the left shows the project structure, including files for 'ec2', 'module', 's3', and 'vpc'.

```
module > main.tf > module "ec2" > instance_type
1 #calling modules
2 module "ec2" {
3   source      = "../ec2"
4   for_each    = var.ec2
5   ami         = each.value["ami"]
6   instance_type = each.value["instance_type"]
7   subnet_id   = "subnet-0962069167a59d185"
8 }
9
10 module "s3" {
11   source      = "../s3"
12   bucket_name = var.bucket_name
13 }
14
15 module "vpc" {
16   source      = "../vpc"
17   tae_name1   = var.tae_name
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v4.56.0

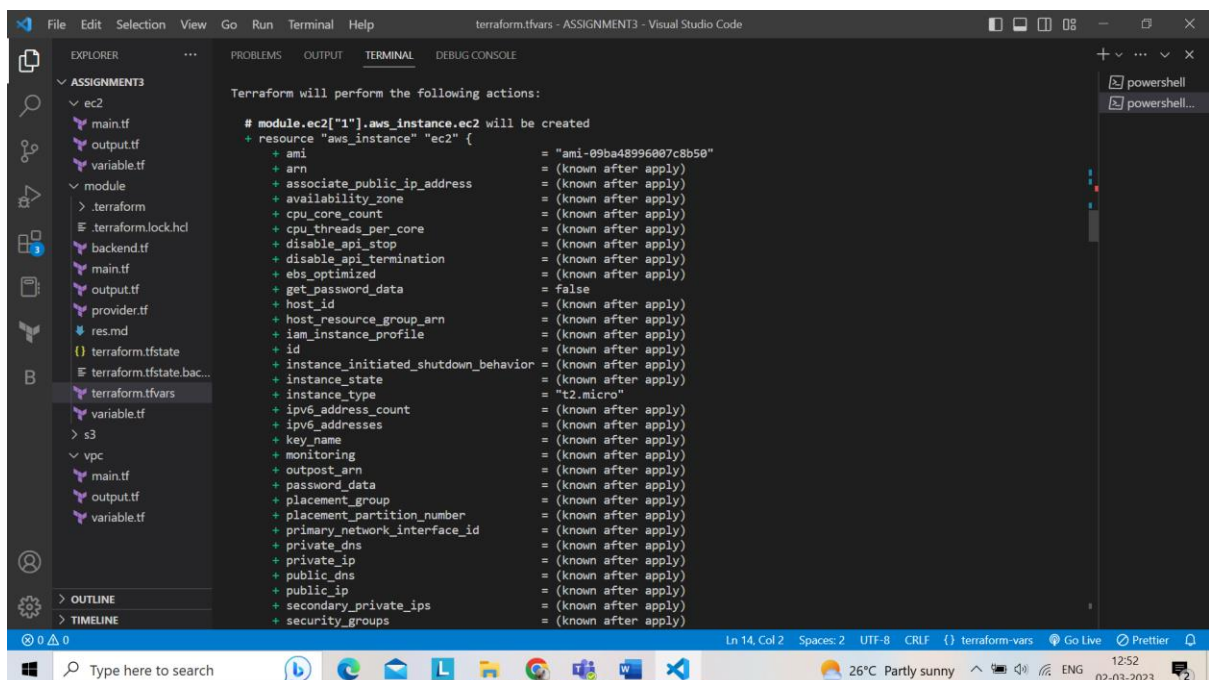
Terraform has been successfully initialized!

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.

PS C:\Users\lenovo\terraform_1.3.7_windows_386\ASSIGNMENT3> module>

CREATING EC2 INSTANCE WITH FOR-EACH LOOP



The screenshot shows the Visual Studio Code interface with the 'terraform.tfvars' file open. The terminal window at the bottom displays the output of the 'terraform plan' command, showing the actions Terraform will perform to create an EC2 instance. The Explorer sidebar on the left shows the project structure, including files for 'ec2', 'module', 's3', and 'vpc'.

```
Terraform will perform the following actions:
# module.ec2["1"].aws_instance.ec2 will be created
+ resource "aws_instance" "ec2" {
  + ami              = "ami-09ba48996007c8b50"
  + arn              = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone = (known after apply)
  + 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)
  + 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_state     = (known after apply)
  + instance_type      = "t2.micro"
  + ipv6_address_count = (known after apply)
  + ipv6_addresses     = (known after apply)
  + key_name           = (known after apply)
  + monitoring         = (known after apply)
  + outpost_arn        = (known after apply)
  + password_data      = (known after apply)
  + placement_group    = (known after apply)
  + placement_partition_number = (known after apply)
  + primary_network_interface_id = (known after apply)
  + private_dns        = (known after apply)
  + private_ip         = (known after apply)
  + public_dns         = (known after apply)
  + public_ip          = (known after apply)
  + secondary_private_ips = (known after apply)
  + security_groups    = (known after apply)
```

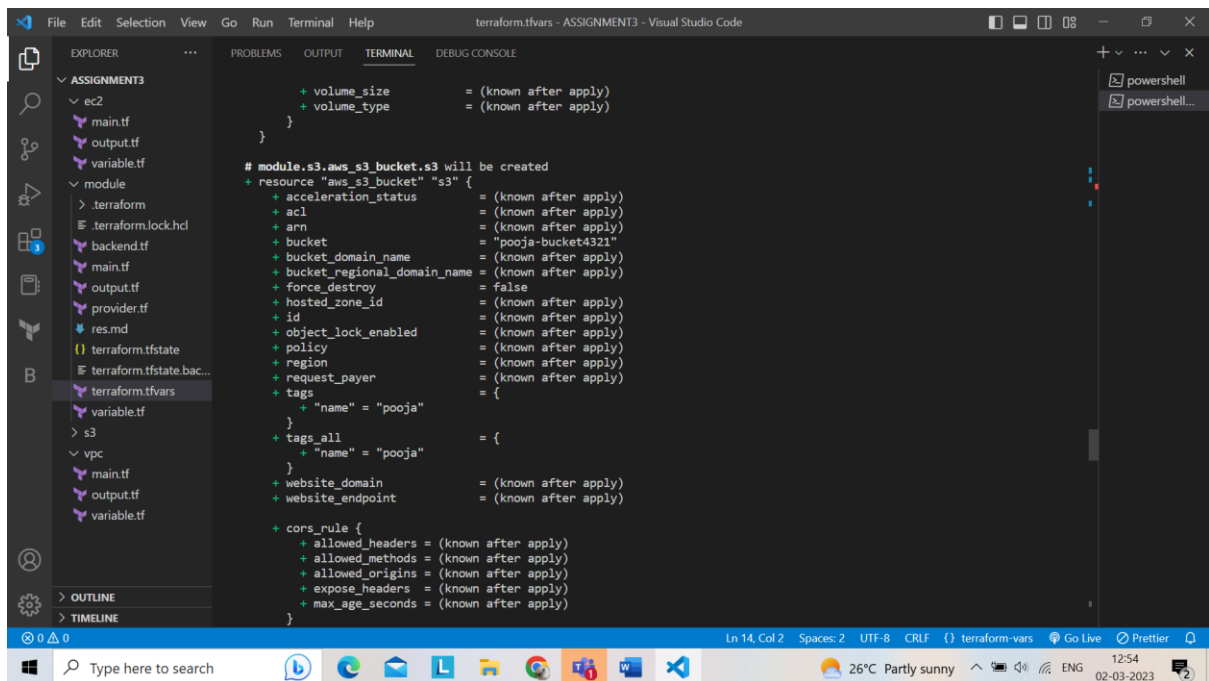
This screenshot shows the Visual Studio Code interface with the Terraform configuration files for an EC2 instance. The Explorer pane on the left shows the project structure, including the `ec2` module and its sub-files. The main editor displays the `main.tf` file, which defines the `aws_instance` resource. The configuration includes tags for the instance, a `capacity_reservation_specification` block, and an `ebs_block_device` block. The status bar at the bottom indicates the file is at line 14, column 2, and the workspace is named `terraform.tfvars - ASSIGNMENT3`.

```
source_dest_check = true
subnet_id         = "subnet-0962069167a59d185"
tags              = {
  "Owner"   = "pooja@cloudq.com"
  "Purpose" = "training"
  "name"    = "pooja-ec2"
}
tags_all          = {
  "Owner"   = "pooja@cloudq.com"
  "Purpose" = "training"
  "name"    = "pooja-ec2"
}
tenancy           = (known after apply)
user_data         = (known after apply)
user_data_base64  = (known after apply)
user_data_replace_on_change = false
volume_tags       = {
  "Name"   = "pooja"
  "Owner"  = "pooja@cloudq.com"
  "Purpose" = "training"
}
vpc_security_group_ids = (known after apply)
capacity_reservation_specification {
  capacity_reservation_preference = (known after apply)
  capacity_reservation_target {
    capacity_reservation_id    = (known after apply)
    capacity_reservation_resource_group_arn = (known after apply)
  }
}
ebs_block_device {
  delete_on_termination = (known after apply)
  device_name           = (known after apply)
  encrypted              = (known after apply)
}
```

This screenshot shows the Visual Studio Code interface displaying the Terraform plan output. The Explorer pane on the left shows the project structure, including the `ec2` module and its sub-files. The main editor displays the `main.tf` file, which defines the `aws_instance` resource. The status bar at the bottom indicates the file is at line 14, column 2, and the workspace is named `terraform.tfvars - ASSIGNMENT3`.

```
# module.ec2["2"].aws_instance.ec2 will be created
+ resource "aws_instance" "ec2" {
  ami              = "ami-09ba48996007c8b50"
  arn              = (known after apply)
  associate_public_ip_address = (known after apply)
  availability_zone = (known after apply)
  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)
  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_state    = (known after apply)
  instance_type      = "t2.micro"
  ipv6_address_count = (known after apply)
  ipv6_addresses     = (known after apply)
  key_name           = (known after apply)
  monitoring         = (known after apply)
  outpost_arn        = (known after apply)
  password_data      = (known after apply)
  placement_group    = (known after apply)
  placement_partition_number = (known after apply)
  primary_network_interface_id = (known after apply)
  private_dns        = (known after apply)
  private_ip         = (known after apply)
  public_dns         = (known after apply)
  public_ip          = (known after apply)
  secondary_private_ips = (known after apply)
  security_groups     = (known after apply)
  source_dest_check   = true
  subnet_id          = "subnet-0962069167a59d185"
}
```

CREATING S3 BUCKET



The screenshot shows the Visual Studio Code interface with the Terraform configuration file `terraform.tfvars` open. The configuration is for an AWS S3 bucket named `s3`. The `resource "aws_s3_bucket" "s3"` block includes various attributes such as `acceleration_status`, `acl`, `arn`, `bucket`, `bucket_domain_name`, `bucket_regional_domain_name`, `force_destroy`, `hosted_zone_id`, `id`, `object_lock_enabled`, `policy`, `region`, `request_payer`, `tags`, `tags_all`, `website_domain`, `website_endpoint`, and `cors_rule`. The `tags` block is defined with `"name" = "pooja"`. The `tags_all` block is also defined with `"name" = "pooja"`. The `cors_rule` block is defined with `allowed_headers`, `allowed_methods`, `allowed_origins`, `expose_headers`, and `max_age_seconds`. The status bar at the bottom indicates the file is at line 14, column 2, with 2 spaces, UTF-8 encoding, and CRLF line endings. The system tray shows a temperature of 26°C, partly sunny weather, and the date 02-03-2023.

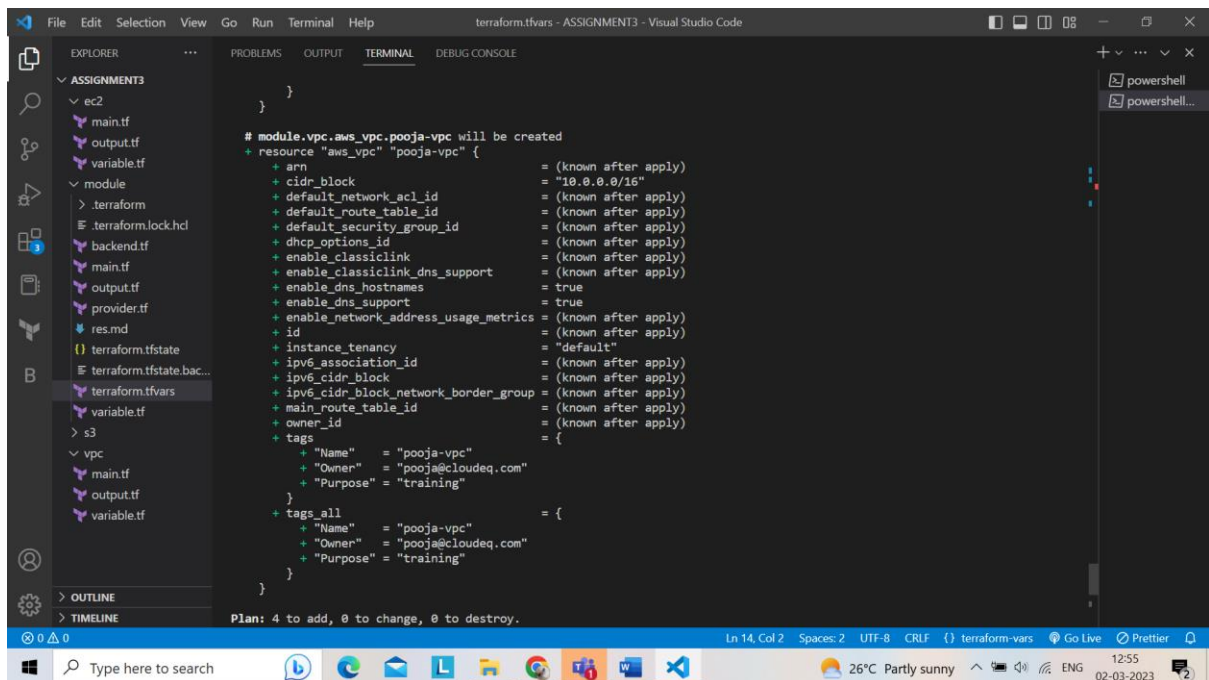
```
File Edit Selection View Go Run Terminal Help terraform.tfvars - ASSIGNMENT3 - Visual Studio Code

EXPLORER PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
ASSIGNMENT3
ec2
main.tf
output.tf
variable.tf
module
.terraform
.terraform.lock.hcl
backend.tf
main.tf
output.tf
provider.tf
res.md
terraform.tfstate
terraform.tfstate.bac...
terraform.tfvars
variable.tf
s3
vpc
main.tf
output.tf
variable.tf

# module.s3.aws_s3_bucket.s3 will be created
+ resource "aws_s3_bucket" "s3" {
+   acceleration_status = (known after apply)
+   acl                 = (known after apply)
+   arn                 = (known after apply)
+   bucket              = "pooja-bucket4321"
+   bucket_domain_name = (known after apply)
+   bucket_regional_domain_name = (known after apply)
+   force_destroy       = false
+   hosted_zone_id      = (known after apply)
+   id                  = (known after apply)
+   object_lock_enabled = (known after apply)
+   policy              = (known after apply)
+   region              = (known after apply)
+   request_payer       = (known after apply)
+   tags                = {
+     "name" = "pooja"
+   }
+   tags_all            = {
+     "name" = "pooja"
+   }
+   website_domain      = (known after apply)
+   website_endpoint    = (known after apply)
+   cors_rule {
+     allowed_headers = (known after apply)
+     allowed_methods = (known after apply)
+     allowed_origins = (known after apply)
+     expose_headers  = (known after apply)
+     max_age_seconds = (known after apply)
+   }
+ }

Ln 14, Col 2 Spaces: 2 UTF-8 CRLF terraform-vars Go Live Prettier
26°C Partly sunny 12:54 02-03-2023
```

CREATING VPC



The screenshot shows the Visual Studio Code interface with the Terraform configuration file `terraform.tfvars` open. The configuration is for an AWS VPC named `vpc`. The `resource "aws_vpc" "pooja-vpc"` block includes various attributes such as `arn`, `cidr_block`, `default_network_acl_id`, `default_route_table_id`, `default_security_group_id`, `dhcp_options_id`, `enable_classiclink`, `enable_classiclink_dns_support`, `enable_dns_hostnames`, `enable_dns_support`, `enable_network_address_usage_metrics`, `id`, `instance_tenancy`, `ipv6_association_id`, `ipv6_cidr_block`, `ipv6_cidr_block_network_border_group`, `main_route_table_id`, `owner_id`, `tags`, and `tags_all`. The `tags` block is defined with `"Name" = "pooja-vpc"`, `"Owner" = "pooja@cloudeq.com"`, and `"Purpose" = "training"`. The `tags_all` block is also defined with the same values. The status bar at the bottom indicates the file is at line 14, column 2, with 2 spaces, UTF-8 encoding, and CRLF line endings. The system tray shows a temperature of 26°C, partly sunny weather, and the date 02-03-2023.

```
File Edit Selection View Go Run Terminal Help terraform.tfvars - ASSIGNMENT3 - Visual Studio Code

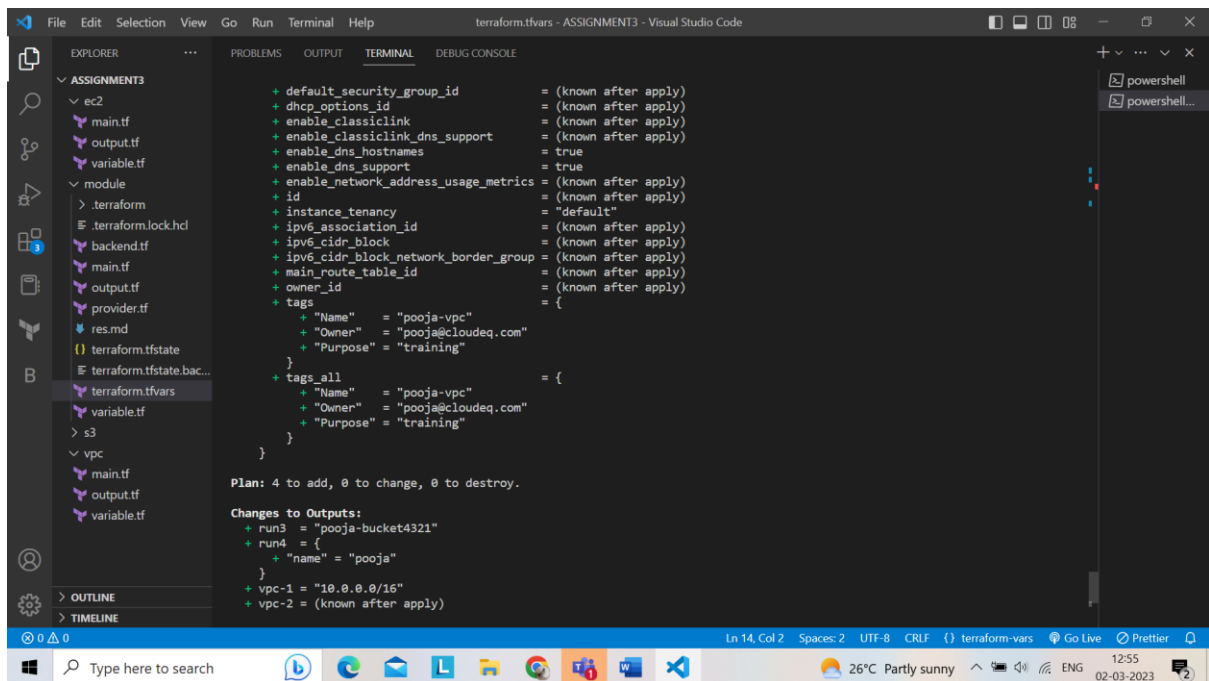
EXPLORER PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
ASSIGNMENT3
ec2
main.tf
output.tf
variable.tf
module
.terraform
.terraform.lock.hcl
backend.tf
main.tf
output.tf
provider.tf
res.md
terraform.tfstate
terraform.tfstate.bac...
terraform.tfvars
variable.tf
s3
vpc
main.tf
output.tf
variable.tf

# module.vpc.aws_vpc.pooja-vpc will be created
+ resource "aws_vpc" "pooja-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_classiclink   = (known after apply)
+   enable_classiclink_dns_support = (known after apply)
+   enable_dns_hostnames = true
+   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" = "pooja-vpc"
+     "Owner" = "pooja@cloudeq.com"
+     "Purpose" = "training"
+   }
+   tags_all            = {
+     "Name" = "pooja-vpc"
+     "Owner" = "pooja@cloudeq.com"
+     "Purpose" = "training"
+   }
+ }

Plan: 4 to add, 0 to change, 0 to destroy.

Ln 14, Col 2 Spaces: 2 UTF-8 CRLF terraform-vars Go Live Prettier
26°C Partly sunny 12:55 02-03-2023
```

OUTPUT



The screenshot shows the Visual Studio Code interface with the 'TERMINAL' tab active. The terminal displays the output of a Terraform plan command. The Explorer sidebar on the left shows a project structure for 'ASSIGNMENT3' with files like 'main.tf', 'output.tf', 'variable.tf', and 'terraform.tfvars'. The terminal output lists various resources and their attributes, including 'default_security_group_id', 'dhcp_options_id', 'enable_classiclink', 'enable_classiclink_dns_support', 'enable_dns_hostnames', 'enable_dns_support', 'enable_network_address_usage_metrics', 'id', 'instance_tenancy', 'ipv6_association_id', 'ipv6_cidr_block', 'ipv6_cidr_block_network_border_group', 'main_route_table_id', 'owner_id', 'tags', and 'tags_all'. It also shows the plan summary: 'Plan: 4 to add, 0 to change, 0 to destroy.' and 'Changes to Outputs:' with values for 'run3', 'run4', 'vpc-1', and 'vpc-2'. The status bar at the bottom indicates 'Ln 14, Col 2', 'Spaces: 2', 'UTF-8', 'CRLF', and 'terraform-vars'.

```
File Edit Selection View Go Run Terminal Help terraform.tfvars - ASSIGNMENT3 - Visual Studio Code

EXPLORER
ASSIGNMENT3
  ec2
    main.tf
    output.tf
    variable.tf
  module
    .terraform
    terraform.lock.hcl
    backend.tf
    main.tf
    output.tf
    provider.tf
    res.md
    terraform.tfstate
    terraform.tfstate.bac...
    terraform.tfvars
    variable.tf
  s3
  vpc
    main.tf
    output.tf
    variable.tf

PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE

+ default_security_group_id = (known after apply)
+ dhcp_options_id          = (known after apply)
+ enable_classiclink        = (known after apply)
+ enable_classiclink_dns_support = (known after apply)
+ enable_dns_hostnames      = true
+ 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" = "pooja-vpc"
+   "Owner" = "pooja@cloudeq.com"
+   "Purpose" = "training"
+ }
+ tags_all                  = {
+   "Name" = "pooja-vpc"
+   "Owner" = "pooja@cloudeq.com"
+   "Purpose" = "training"
+ }

Plan: 4 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ run3 = "pooja-bucket4321"
+ run4 = {
+   "name" = "pooja"
+ }
+ vpc-1 = "10.0.0.0/16"
+ vpc-2 = (known after apply)
```