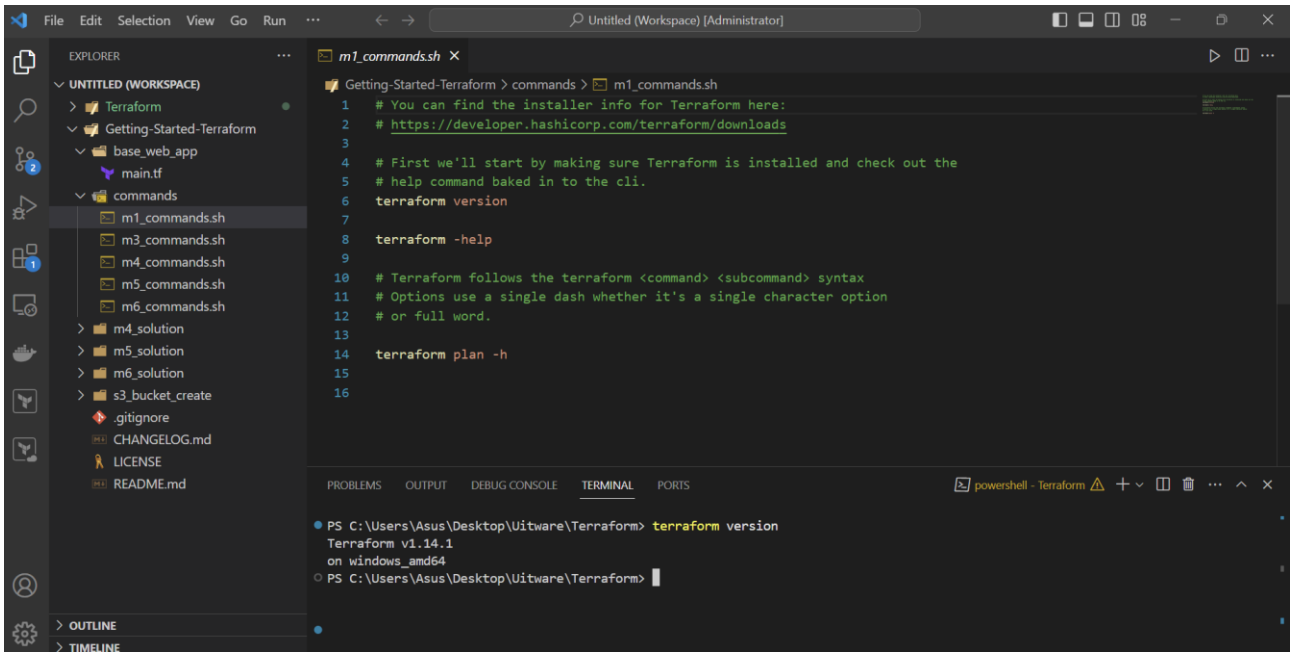
PluralSight Terraform report

Installing Terraform

```
C:\Users\Asus>choco install terraform
Chocolatey v2.3.0
3 validations performed. 2 success(es), 1 warning(s), and 0 error(s).

Validation Warnings:
 - A pending system reboot request has been detected, however, this is
   being ignored due to the current Chocolatey configuration.  If you
   want to halt when this occurs, then either set the global feature
   using:
     choco feature enable --name="exitOnRebootDetected"
   or pass the option --exit-when-reboot-detected.

Installing the following packages:
terraform
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading terraform 1.14.1... 100%

terraform v1.14.1 [Approved]
terraform package files install completed. Performing other installation steps.
The package terraform wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): Y

Removing old terraform plugins
```

```
Removing old terraform plugins
Downloading terraform 64 bit
  from 'https://releases.hashicorp.com/terraform/1.14.1/terraform_1.14.1_windows_amd64.zip'
Progress: 100% - Completed download of C:\Users\Asus\AppData\Local\Temp\chocolatey\terraform\1.14.1\terraform_1.14.1_win
dows_amd64.zip (29.78 MB).
Download of terraform_1.14.1_windows_amd64.zip (29.78 MB) completed.
Hashes match.
Extracting C:\Users\Asus\AppData\Local\Temp\chocolatey\terraform\1.14.1\terraform_1.14.1_windows_amd64.zip to C:\Program
Data\chocolatey\lib\terraform\tools...
C:\ProgramData\chocolatey\lib\terraform\tools
 ShimGen has successfully created a shim for terraform.exe
 The install of terraform was successful.
  Deployed to 'C:\ProgramData\chocolatey\lib\terraform\tools'
```

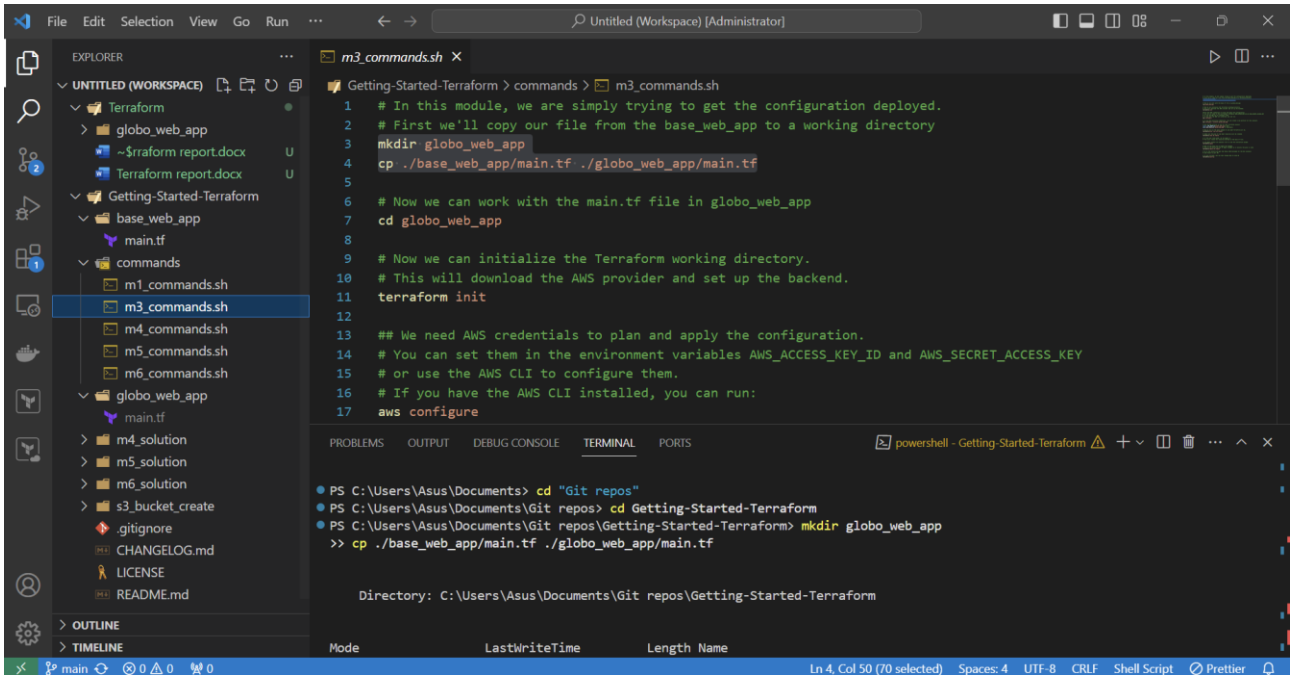
Cloned a repo for practice



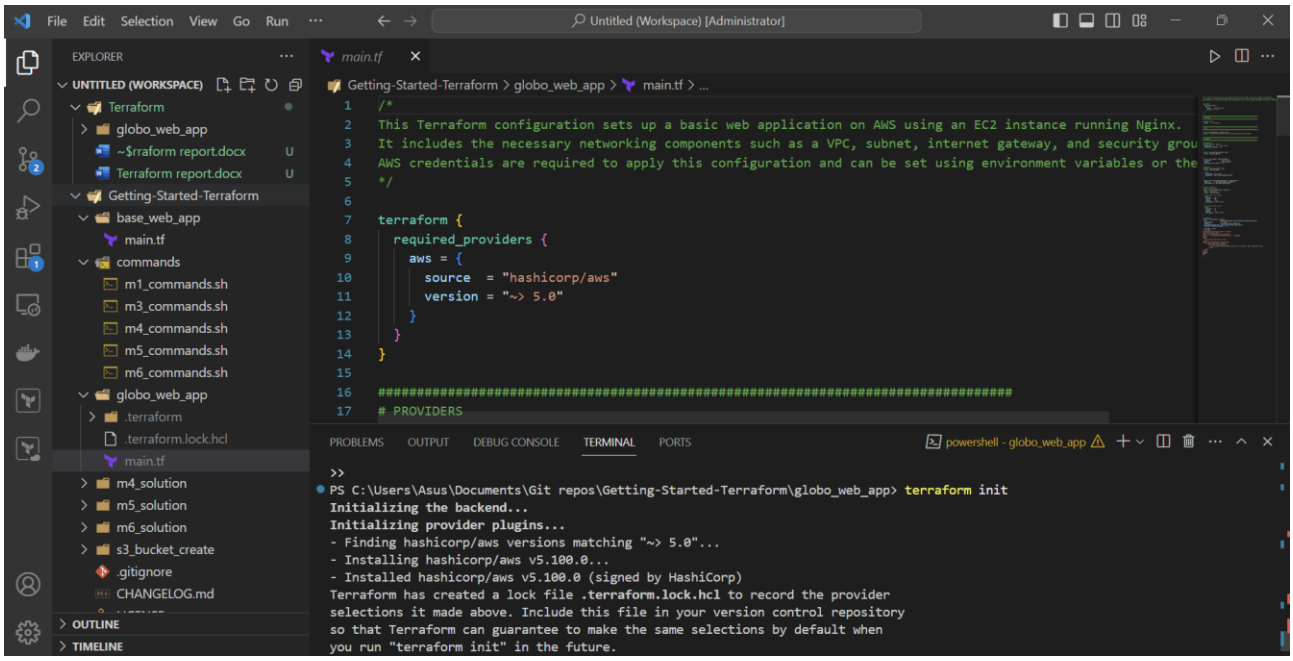
Terraform version

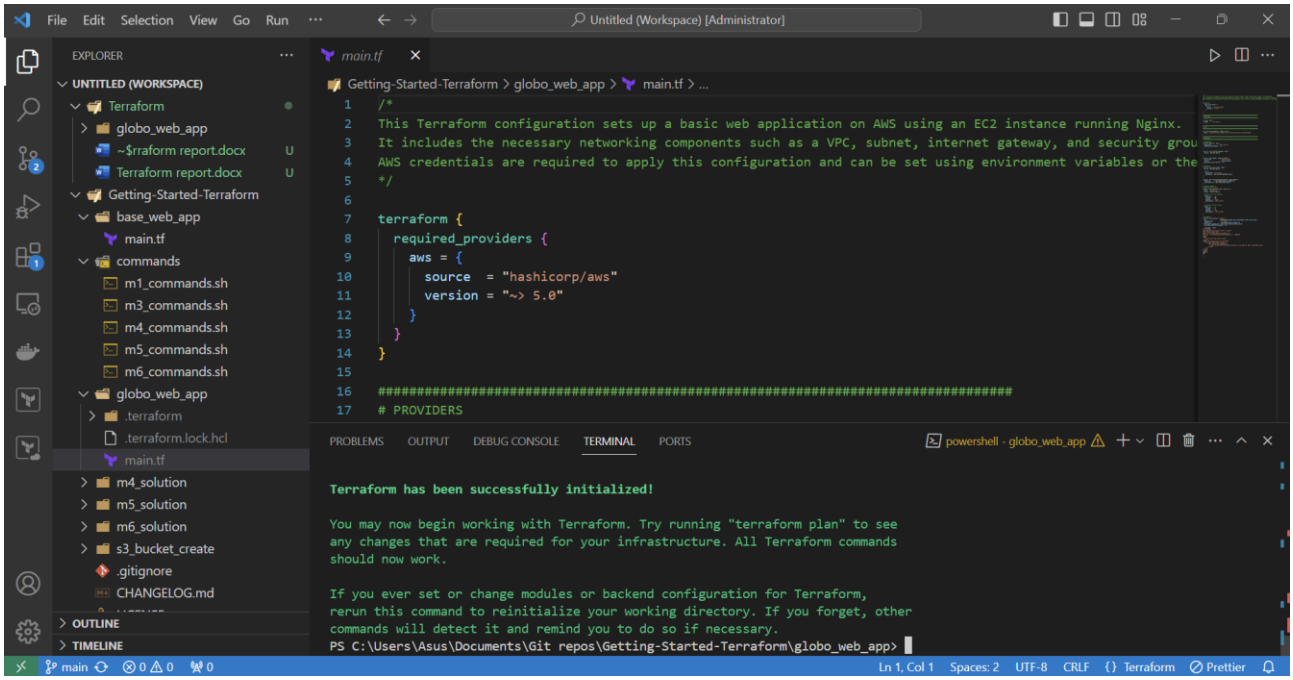
Running terraform init

Creating a folder and copying main file

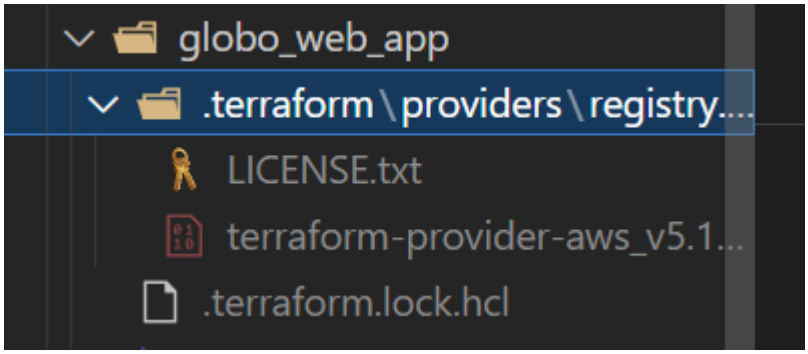


Running terraform init



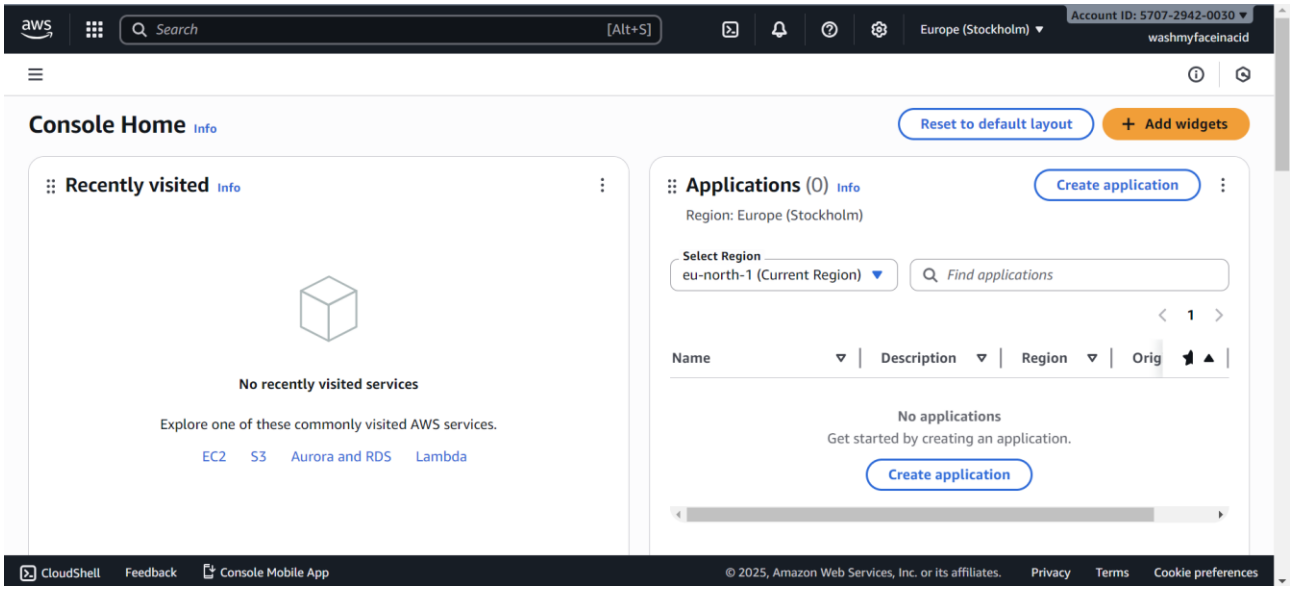


Now we have a file to talk to our provider

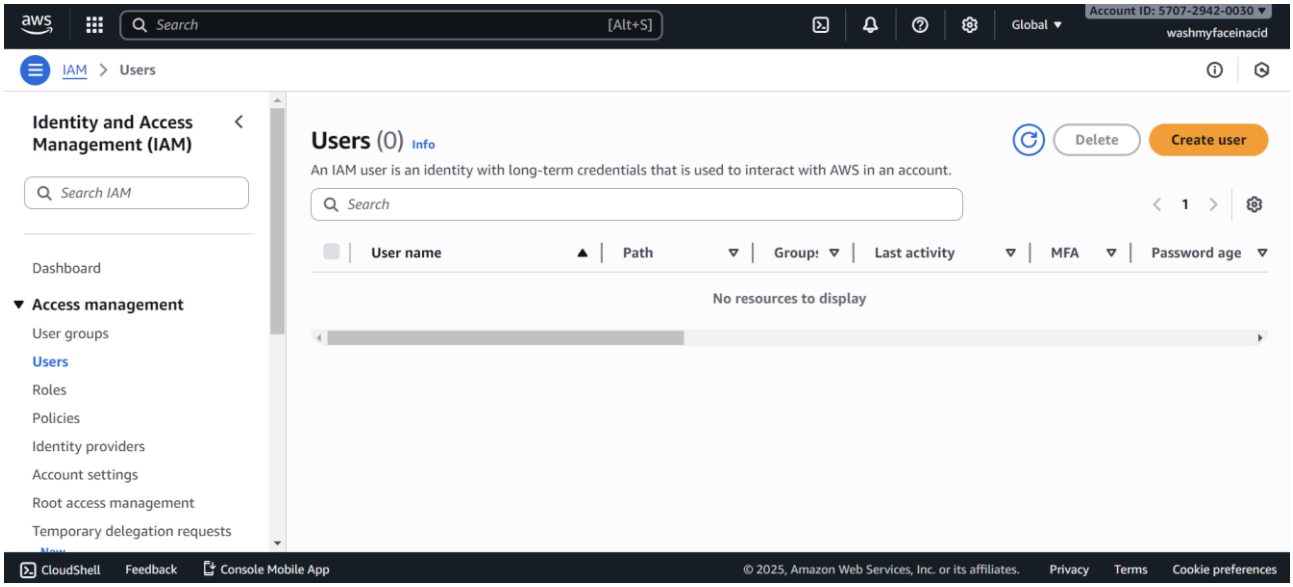


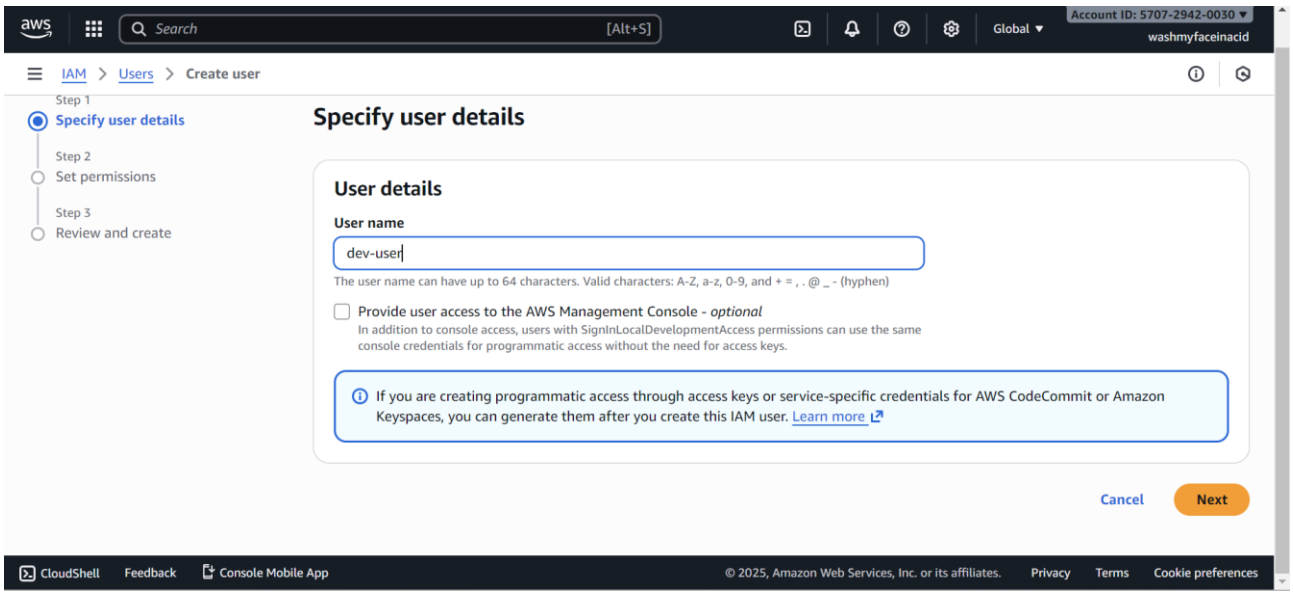
Initialization successful

Also created AWS account :)

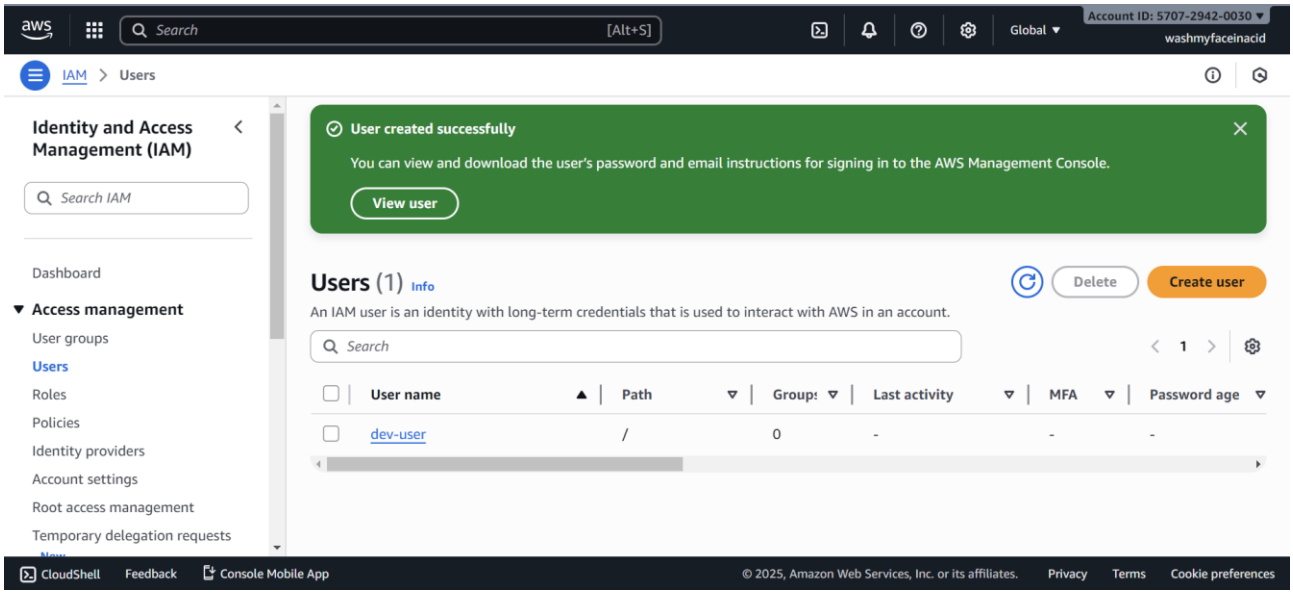


Creating access keys

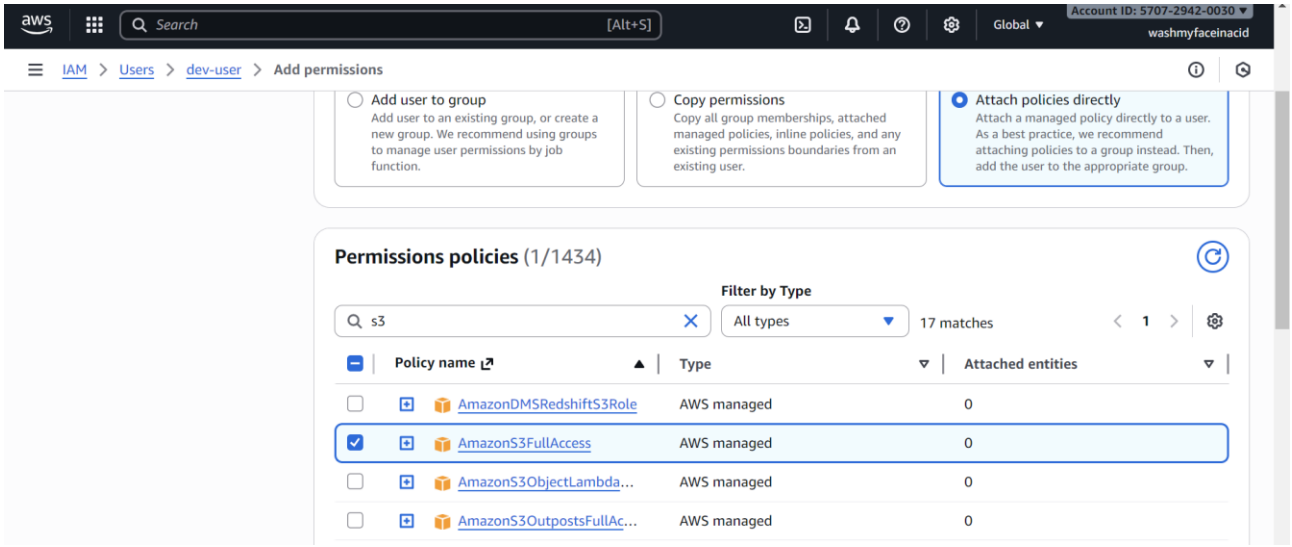




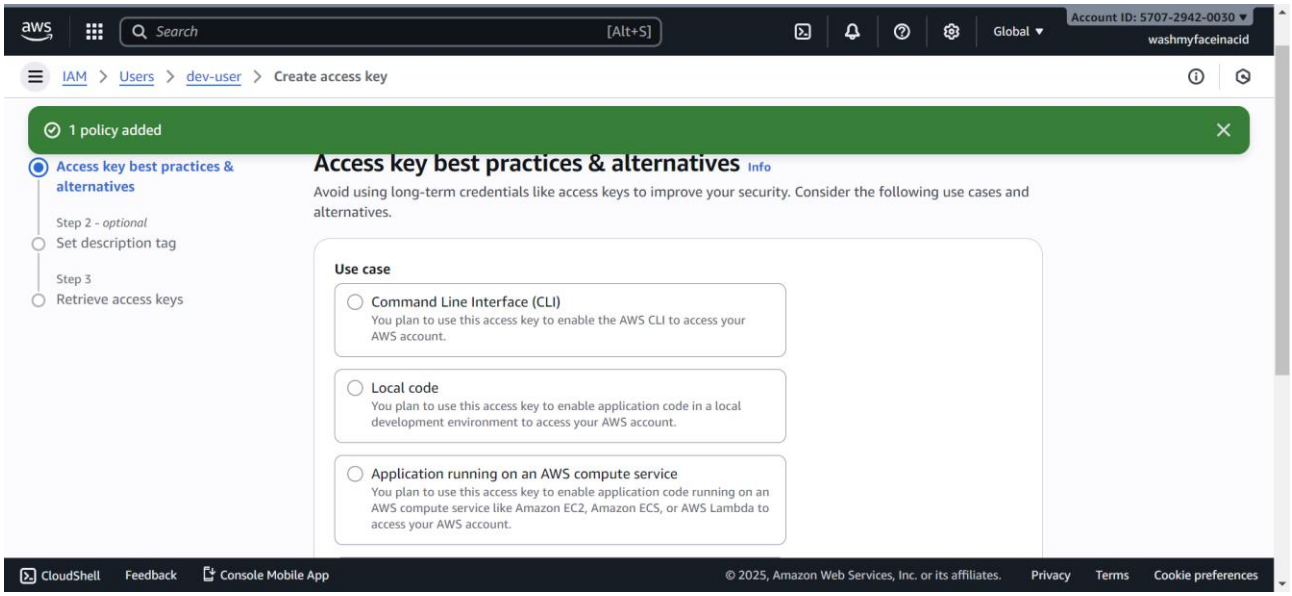
User created

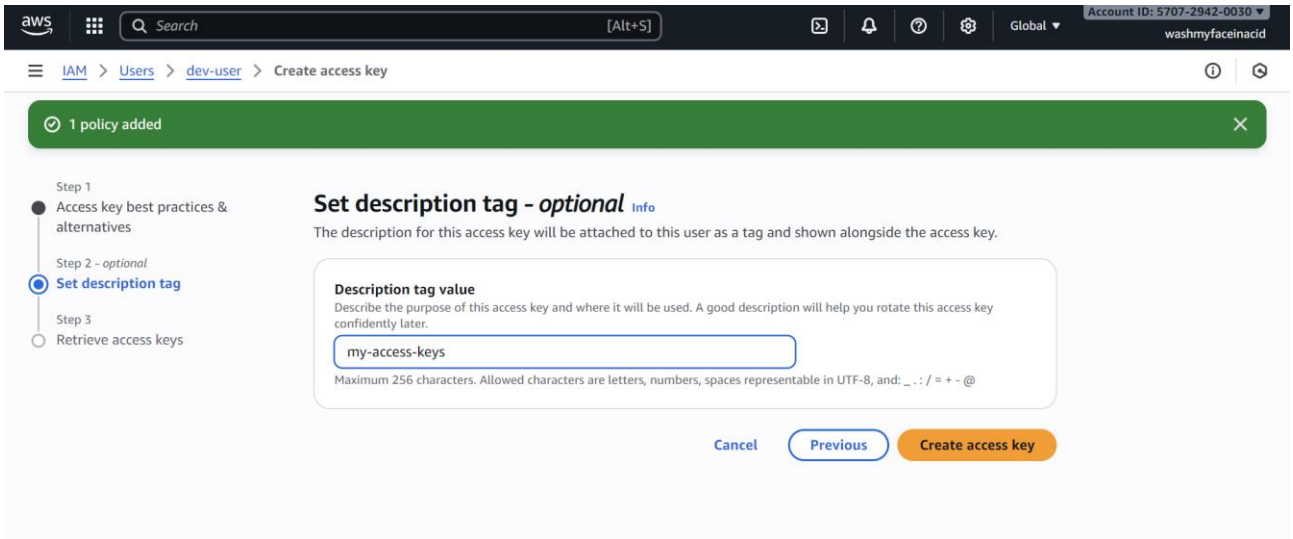


Adding full access

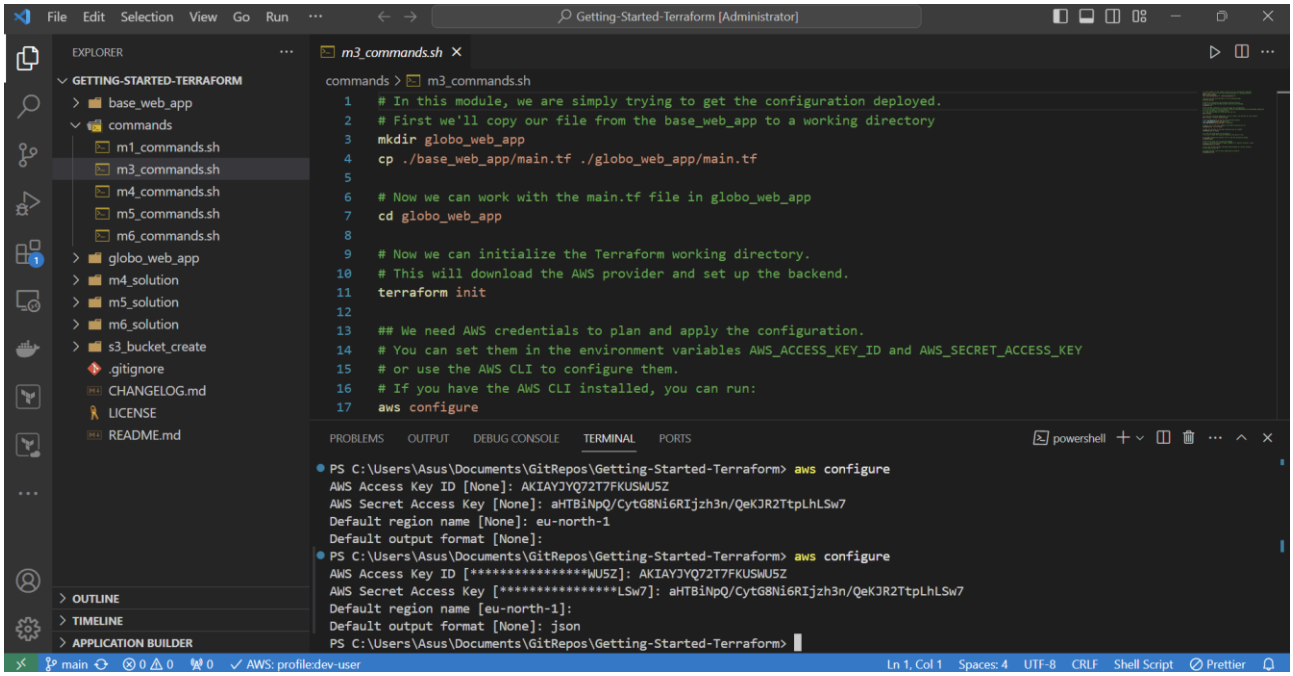


Creating access key

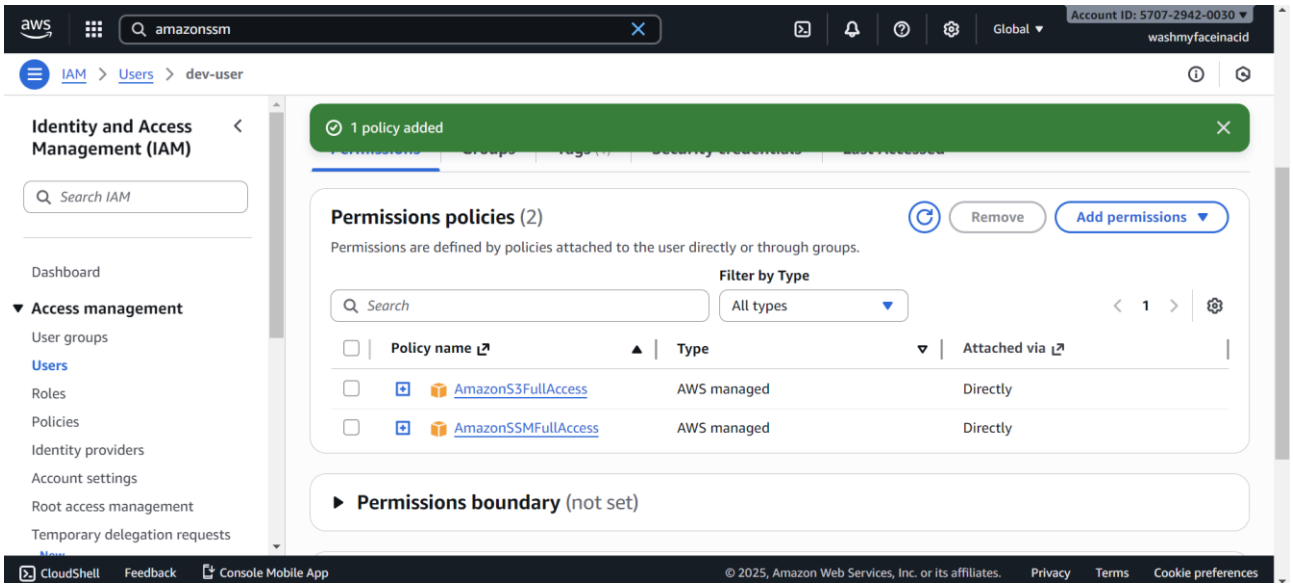




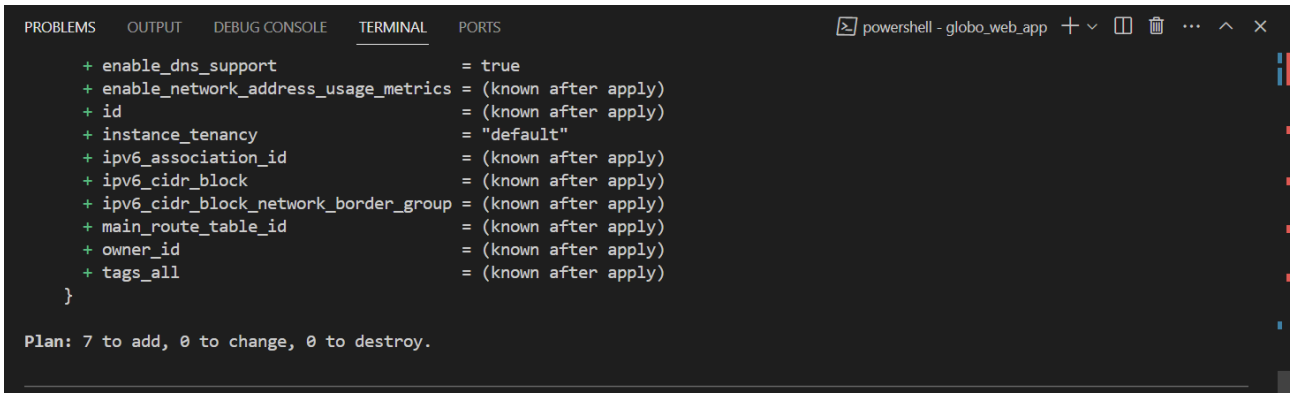
Configured AWS for VScode



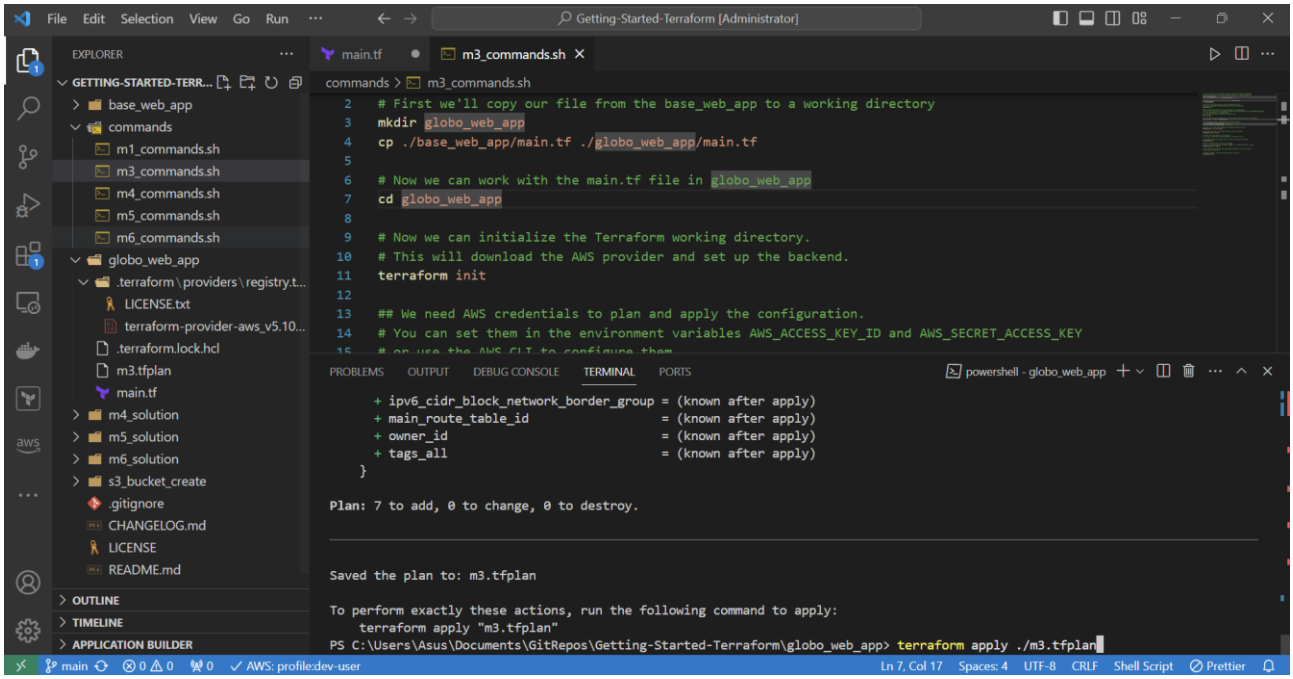
Added SSM permission because of blocking



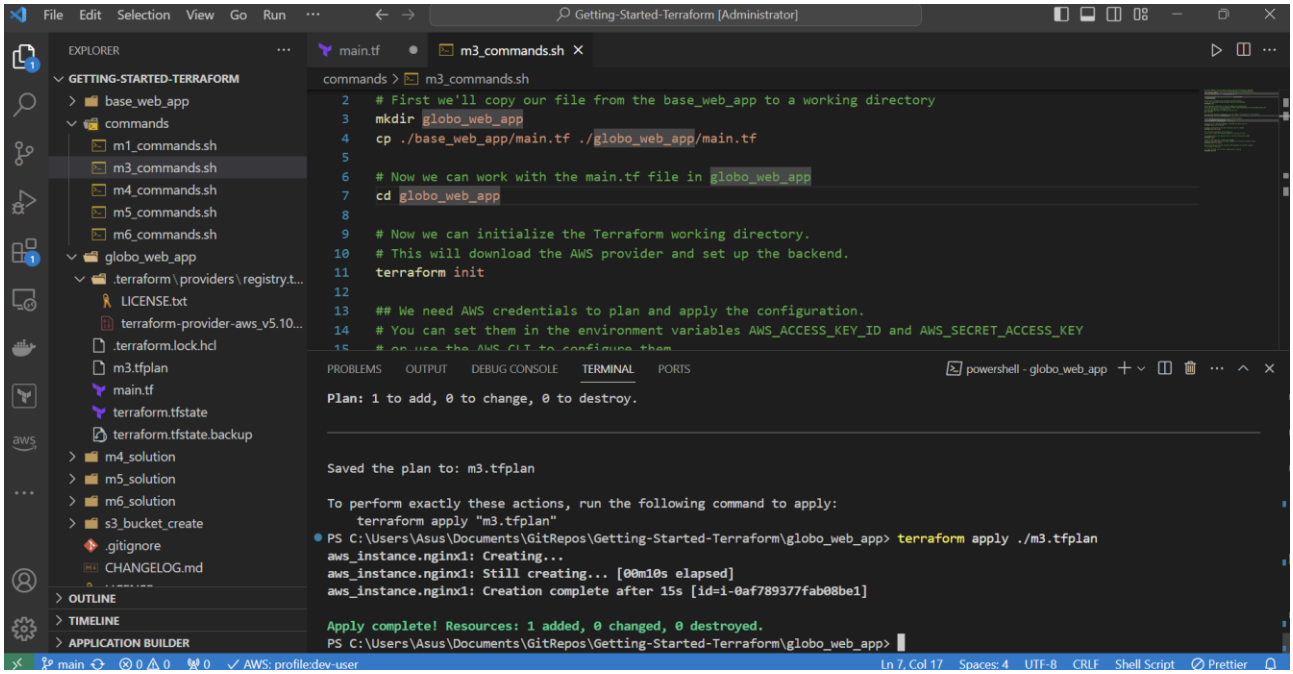
Created



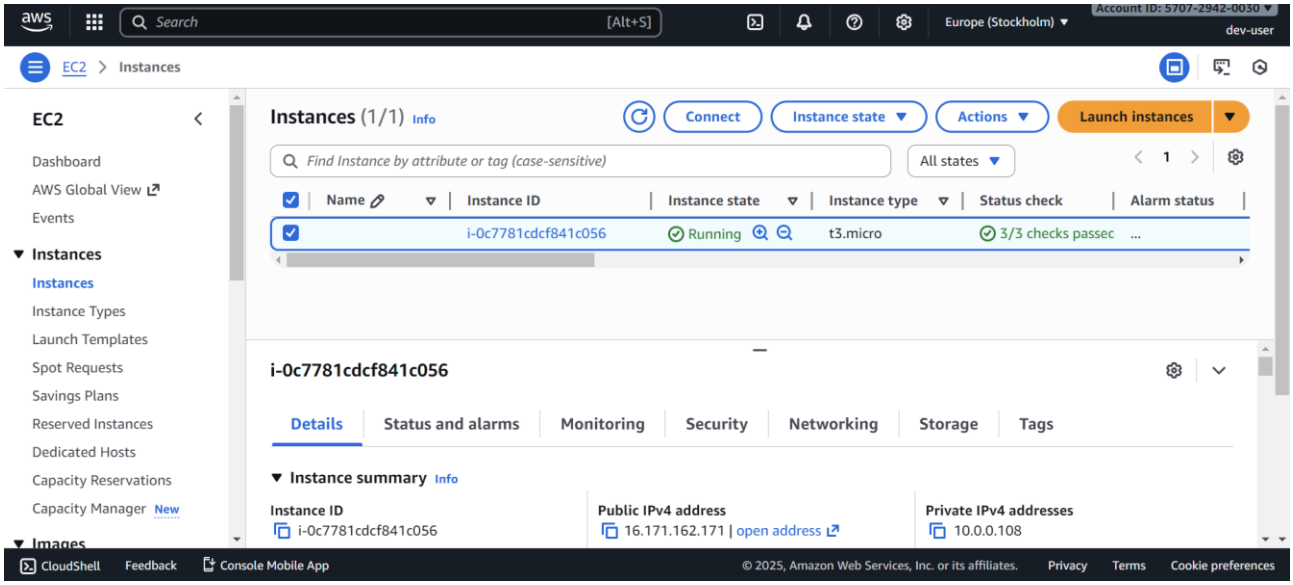
Applying our terraform plan



Applied, deployment completed

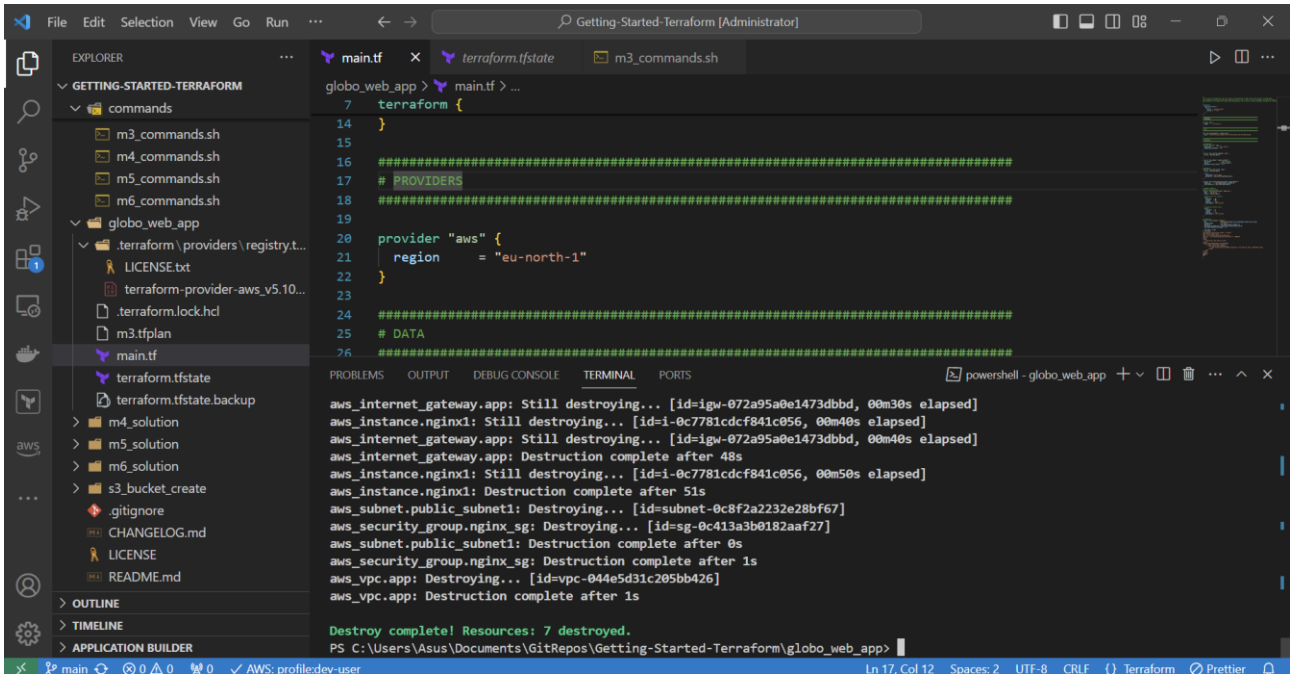


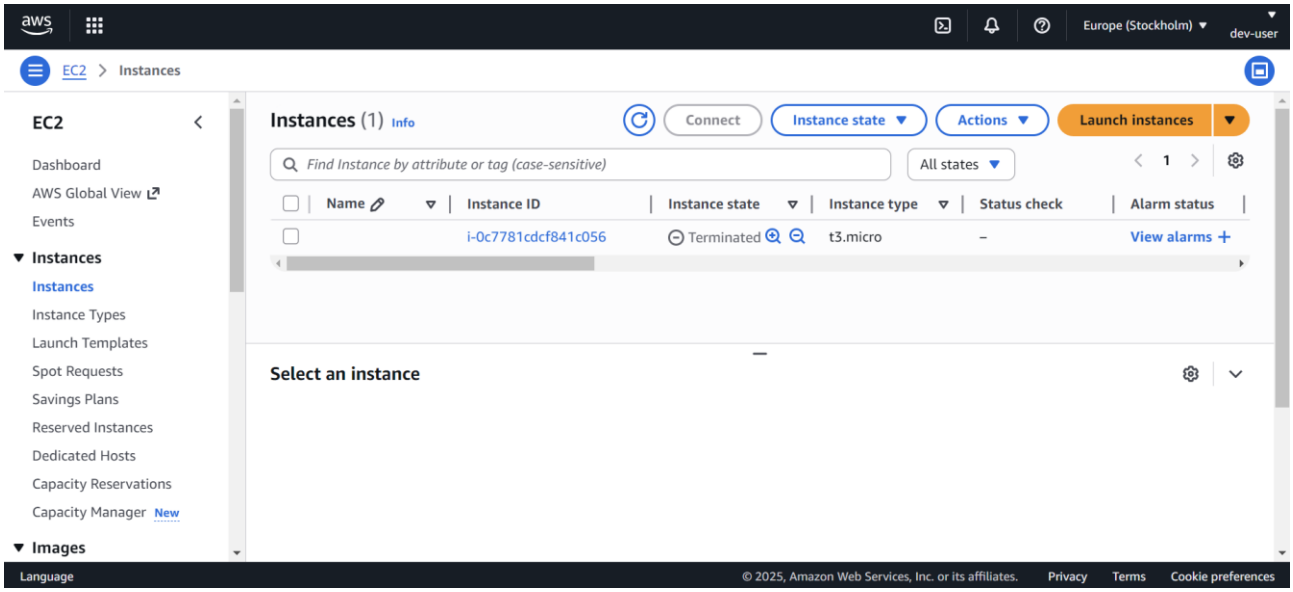
Added



Website:

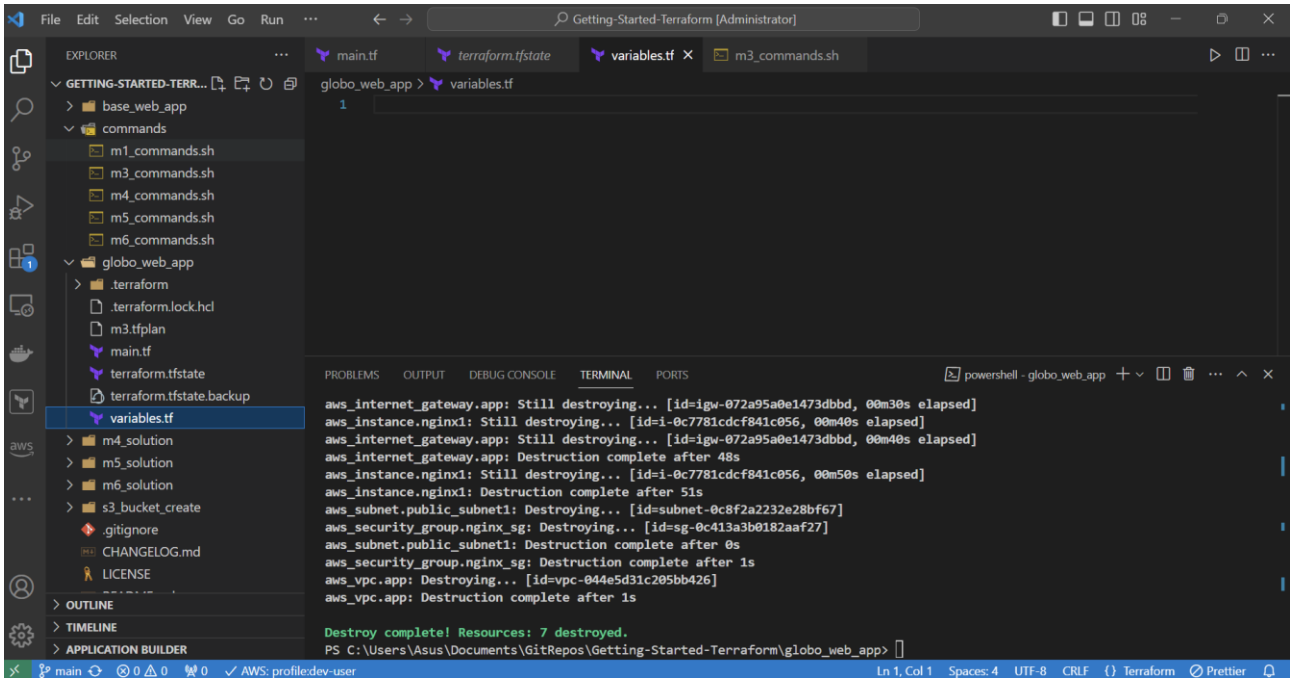
Destroyed



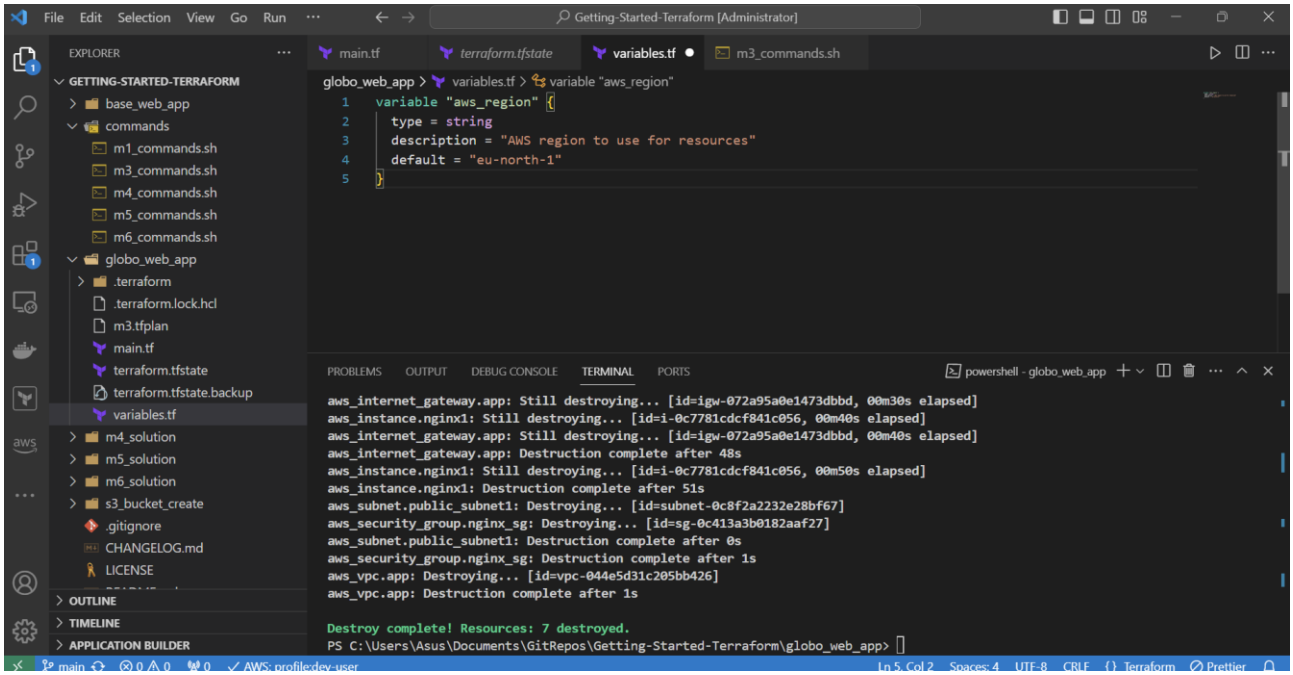


Lets use variables in terraform

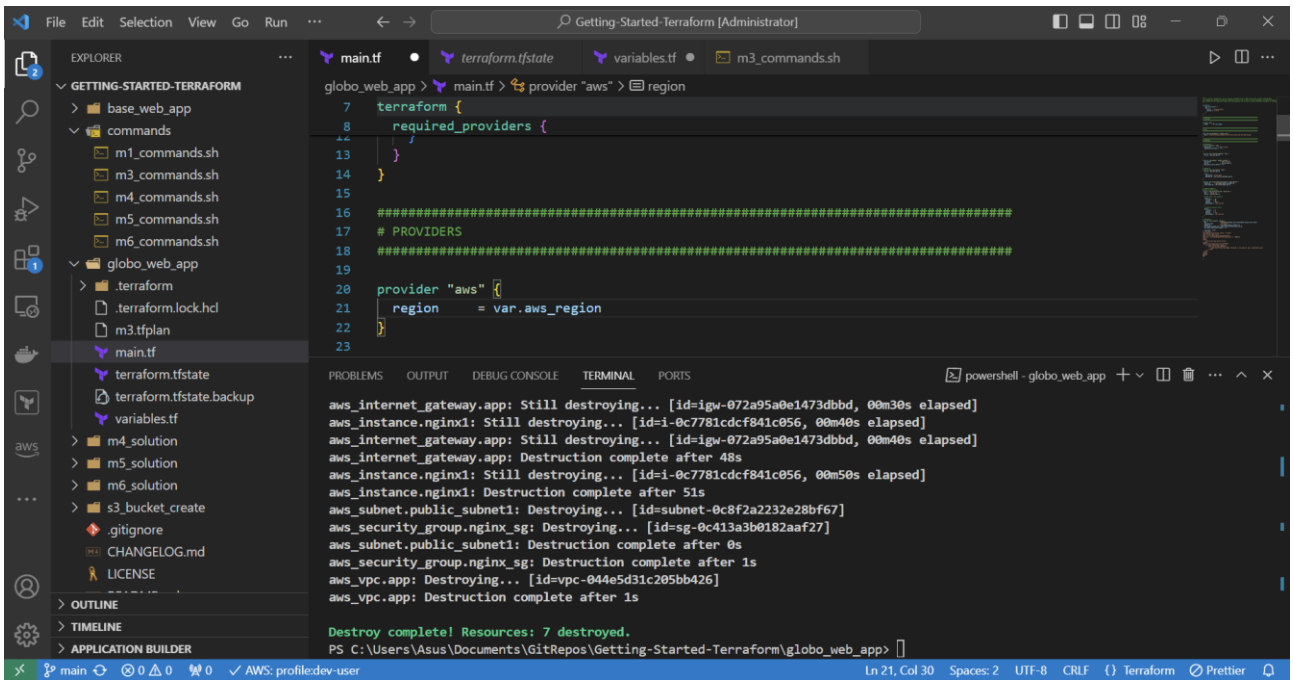
Creating file variables.tf



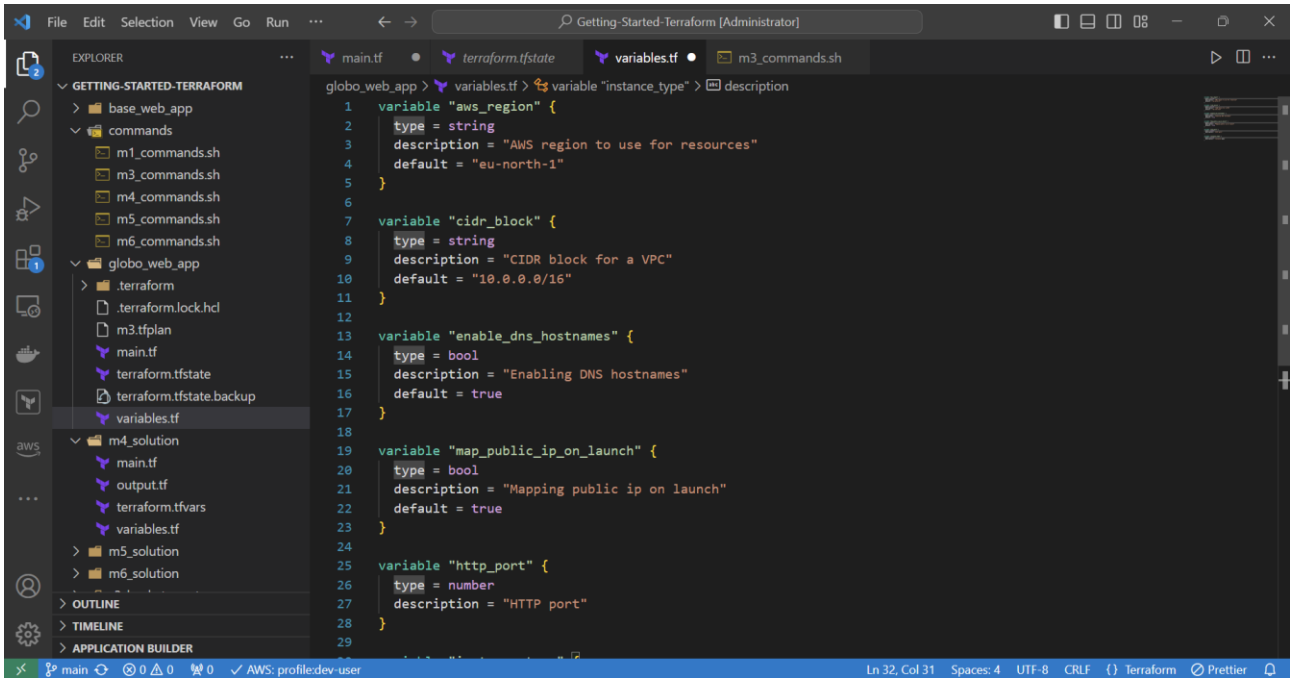
Creating a variable for aws region



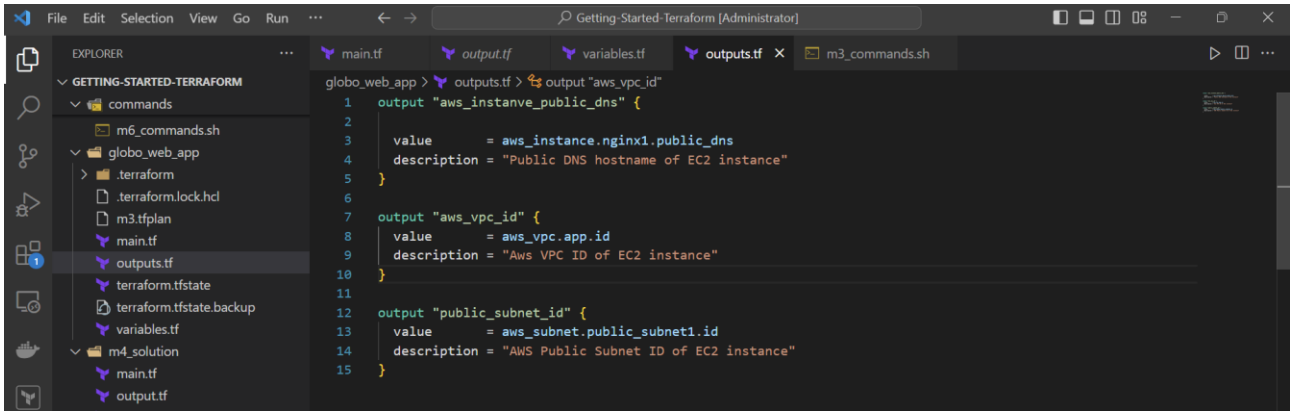
Replacing region with variable:



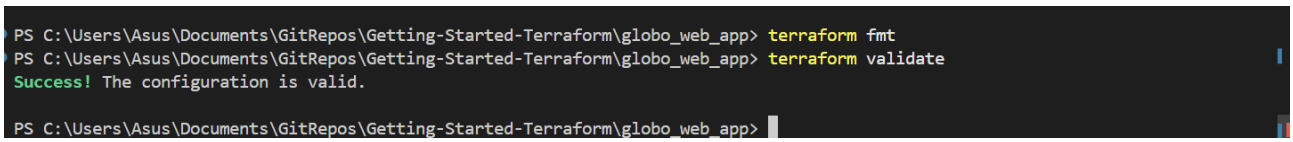
Adding variables to our main.tf file:



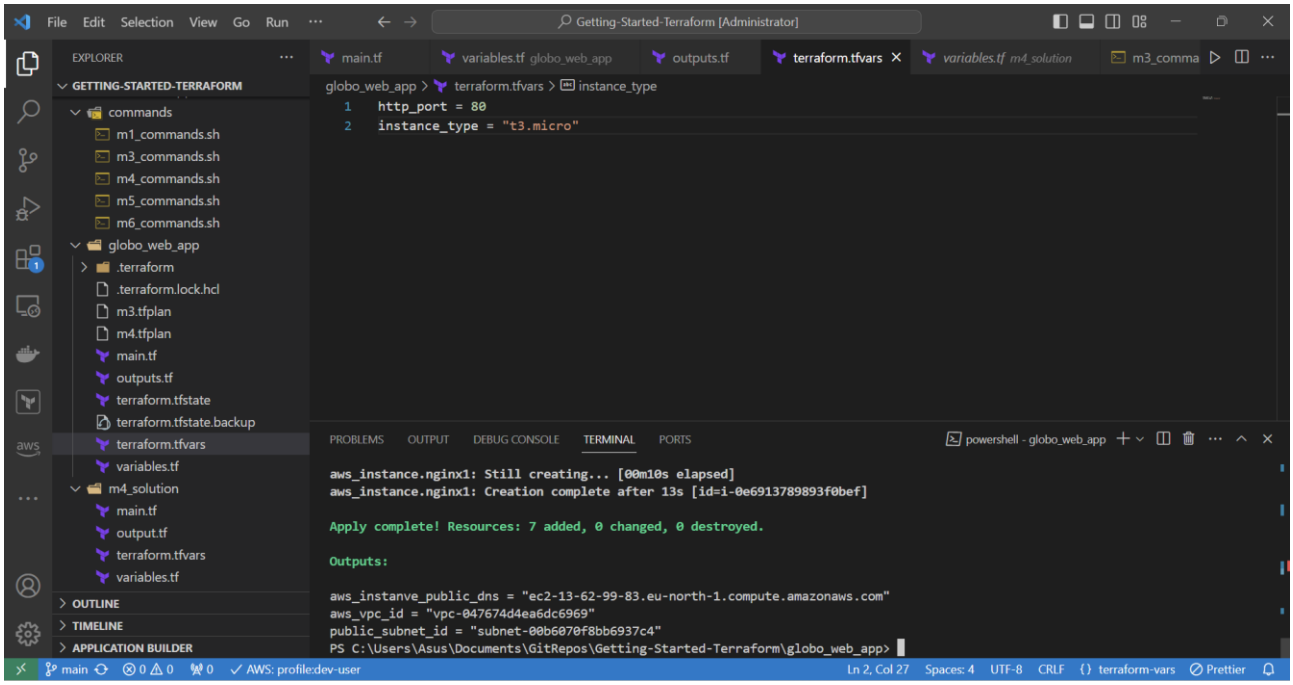
Also adding outputs:



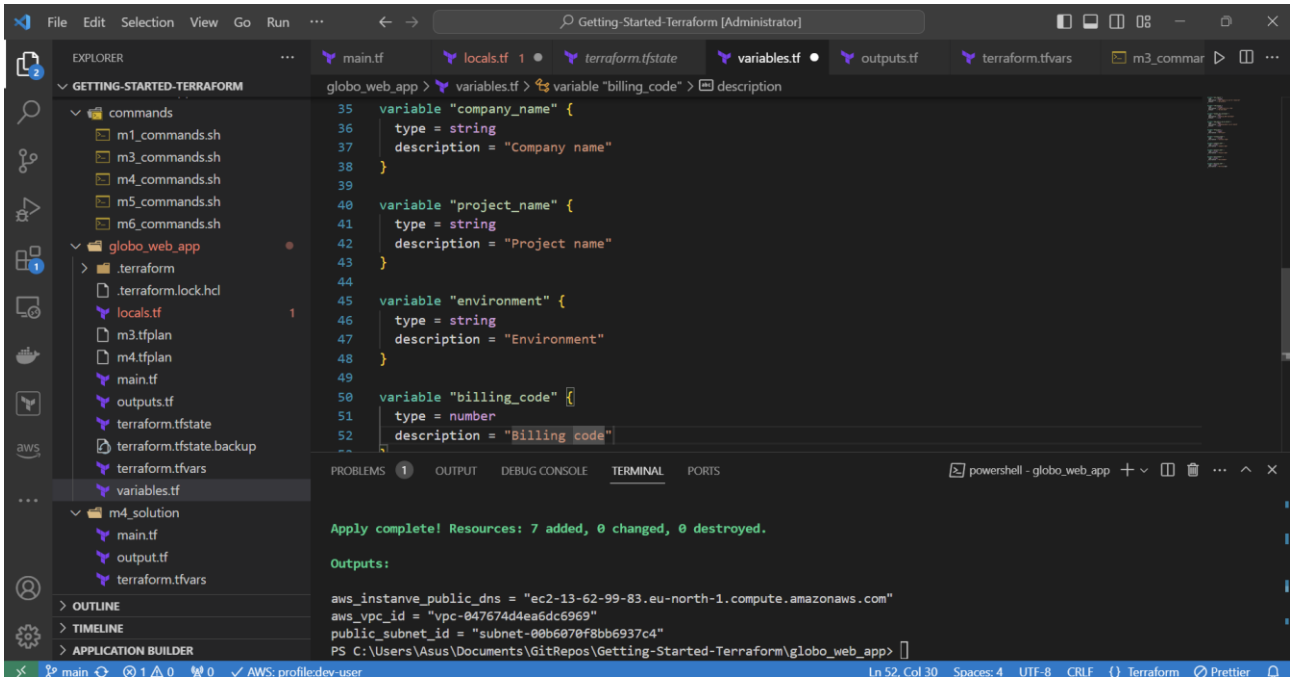
Checking if formatting and syntax and logic is good



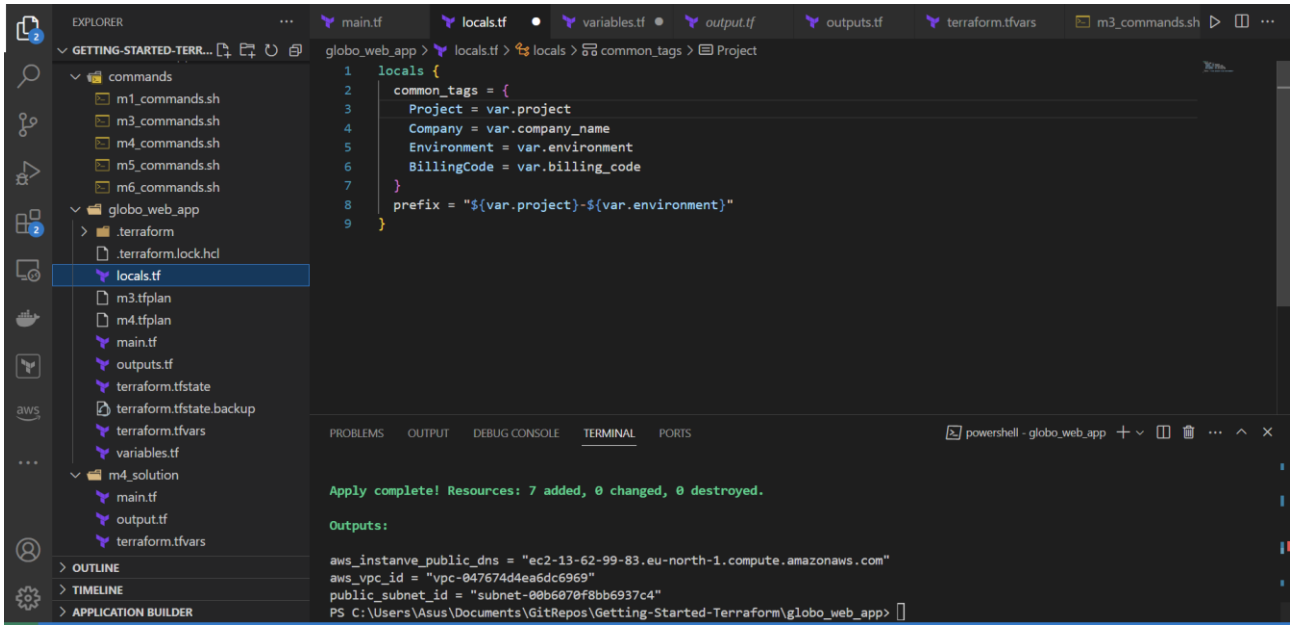
Applying new plan with outputs



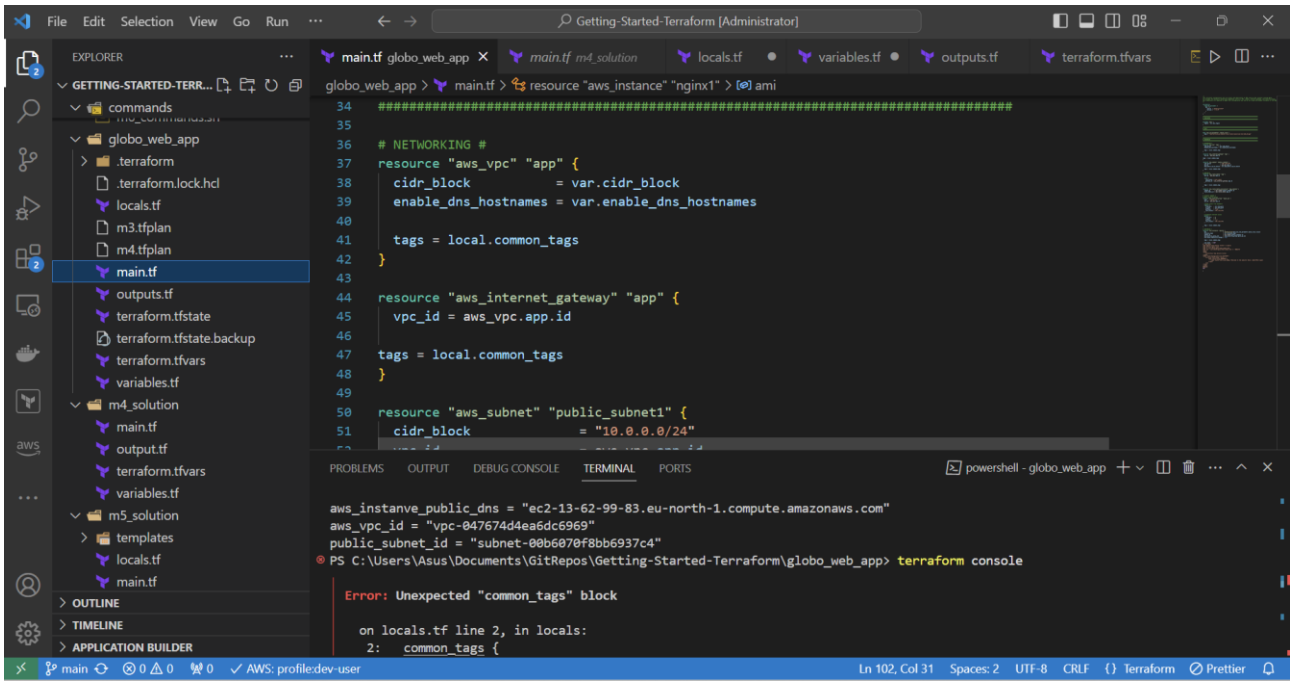
Adding 4 variables

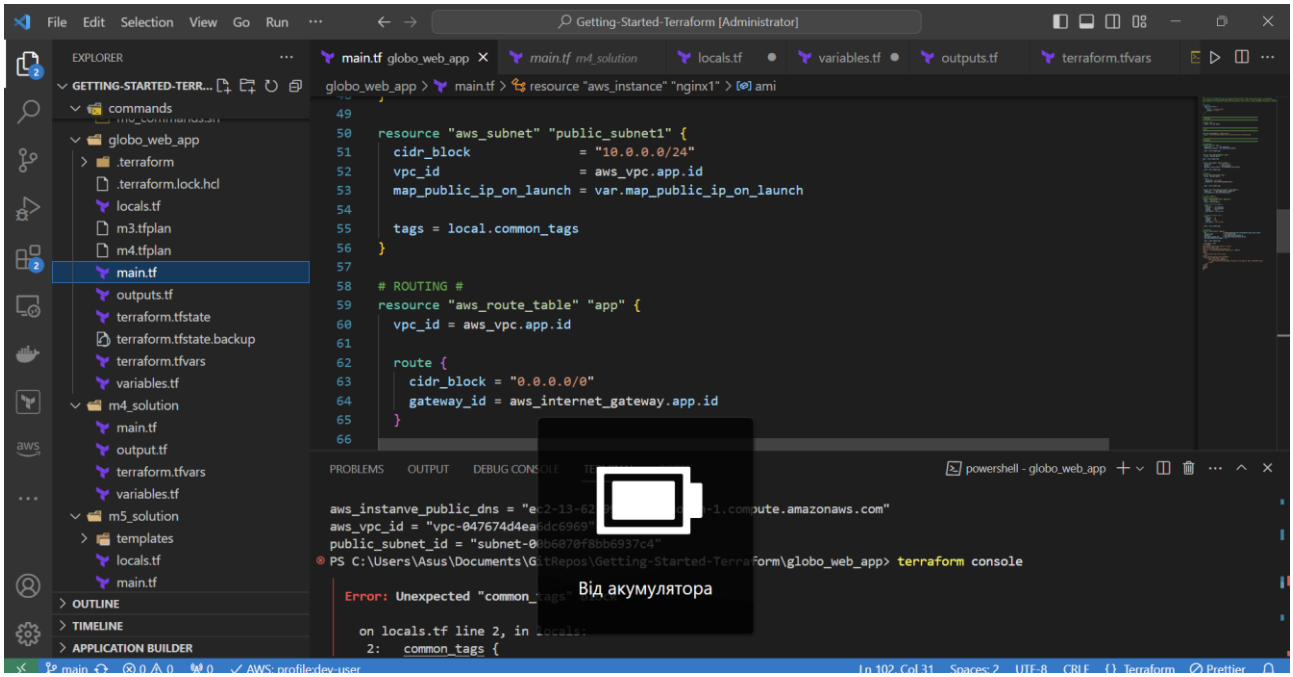


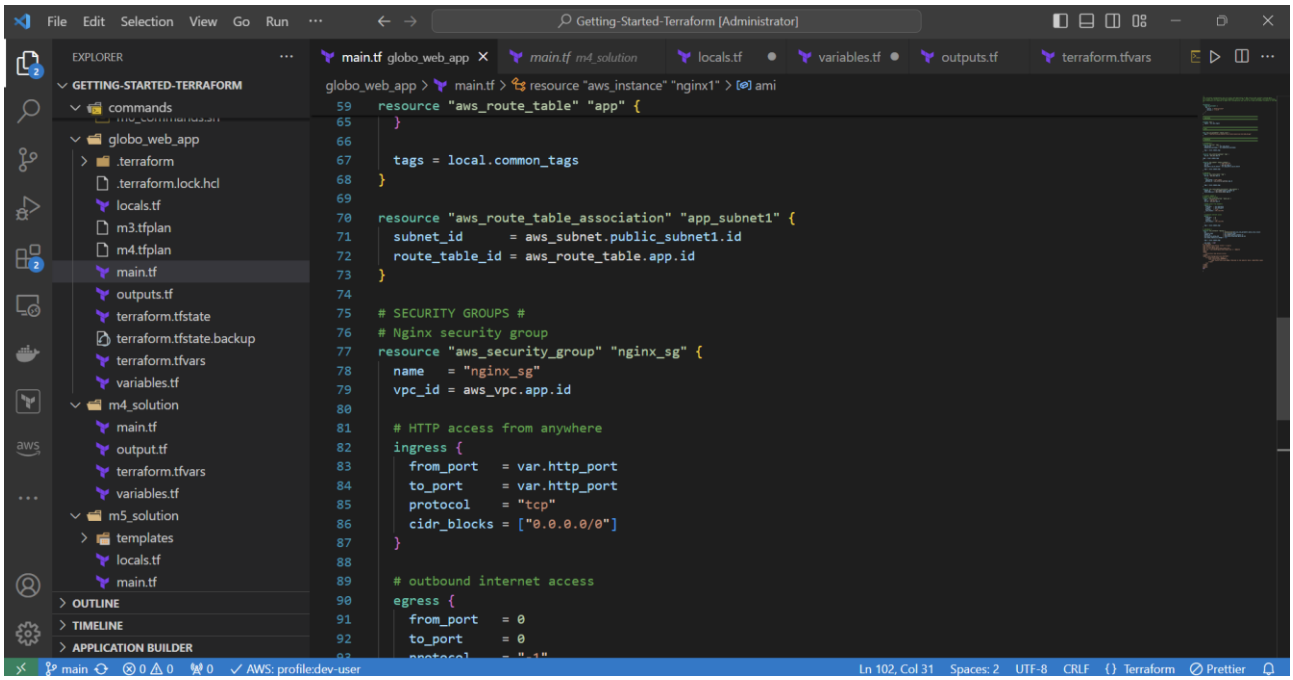
Then add them as locals

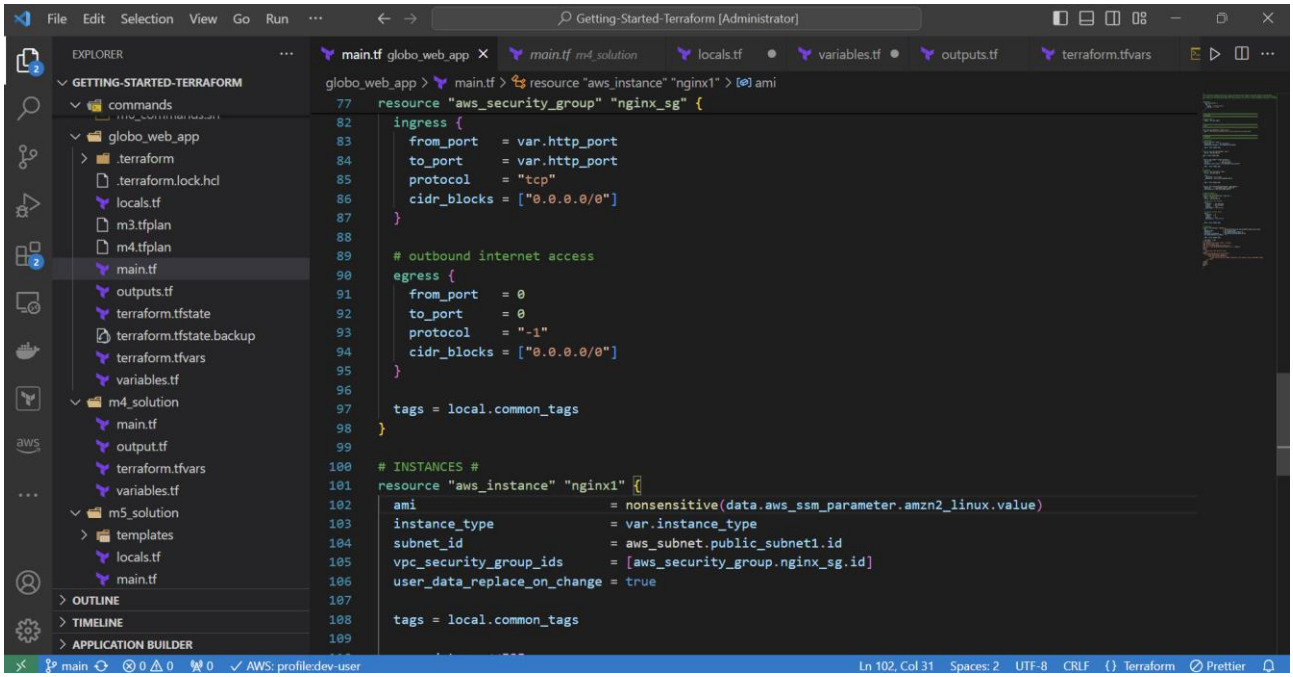


Also added tags to all blocks where I can add tags

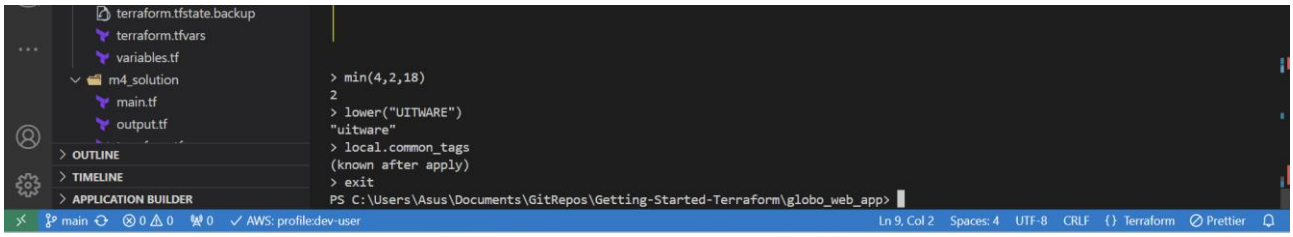



File Edit Selection View Go Run
Getting-Started-Terraform [Administrator]
EXPLORER
GETTING-STARTED-TERR...
commands
globo_web_app
.terraform
.terraform.lock.hcl
locals.tf
m3.tfplan
m4.tfplan
main.tf
outputs.tf
terraform.tfstate
terraform.tfstate.backup
terraform.tfvars
variables.tf
m4_solution
main.tf
output.tf
terraform.tfvars
variables.tf
m5_solution
templates
locals.tf
main.tf
OUTLINE
TIMELINE
APPLICATION BUILDER
main.tf globo_web_app
main.tf m4_solution
locals.tf
variables.tf
outputs.tf
terraform.tfvars
globo_web_app > main.tf > resource "aws_instance" "nginx1" > ami
resource "aws_subnet" "public_subnet1" {
cidr_block = "10.0.0.0/24"
vpc_id = aws_vpc.app.id
map_public_ip_on_launch = var.map_public_ip_on_launch
tags = local.common_tags
}
# ROUTING #
resource "aws_route_table" "app" {
vpc_id = aws_vpc.app.id
route {
cidr_block = "0.0.0.0/0"
gateway_id = aws_internet_gateway.app.id
}
PROBLEMS
OUTPUT
DEBUG CONSOLE
powershell - globo_web_app
aws_instanve_public_dns = "e
ute.amazonaws.com"
aws_vpc_id = "vpc-047674d4ea
public_subnet_id = "subnet-0
PS C:\Users\Asus\Documents\G
form\globo_web_app> terraform console
Error: Unexpected "common_
on locals.tf line 2, in
2: common_tags {
Від акумулятора
main
0 0
0
AWS: profile:dev-user
Ln 102, Col 31
Spaces: 2
UTF-8
CRLF
Terraform
Prettier

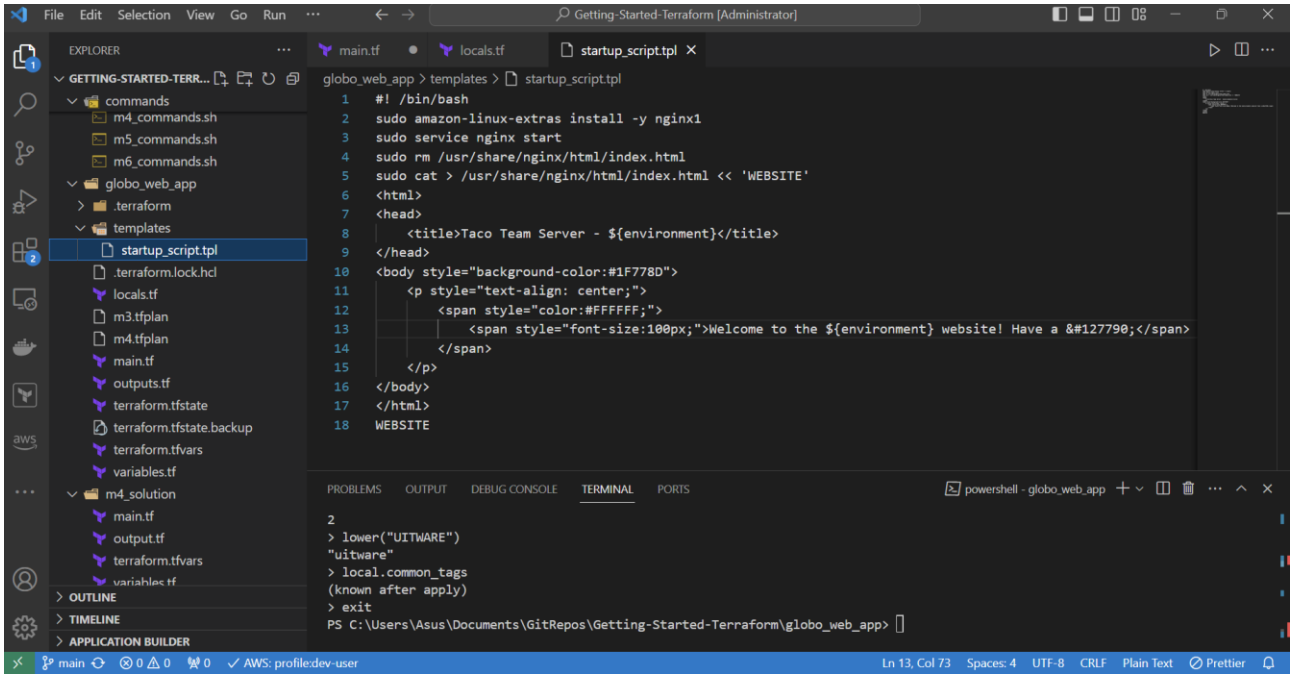

File Edit Selection View Go Run
Getting-Started-Terraform [Administrator]
EXPLORER
GETTING-STARTED-TERRAFORM
commands
globo_web_app
.terraform
.terraform.lock.hcl
locals.tf
m3.tfplan
m4.tfplan
main.tf
outputs.tf
terraform.tfstate
terraform.tfstate.backup
terraform.tfvars
variables.tf
m4_solution
main.tf
output.tf
terraform.tfvars
variables.tf
m5_solution
templates
locals.tf
main.tf
OUTLINE
TIMELINE
APPLICATION BUILDER
main.tf globo_web_app
main.tf m4_solution
locals.tf
variables.tf
outputs.tf
terraform.tfvars
globo_web_app > main.tf > resource "aws_instance" "nginx1" > ami
resource "aws_route_table" "app" {
}
tags = local.common_tags
}
resource "aws_route_table_association" "app_subnet1" {
subnet_id = aws_subnet.public_subnet1.id
route_table_id = aws_route_table.app.id
}
# SECURITY GROUPS #
# Nginx security group
resource "aws_security_group" "nginx_sg" {
name = "nginx_sg"
vpc_id = aws_vpc.app.id
# HTTP access from anywhere
ingress {
from_port = var.http_port
to_port = var.http_port
protocol = "tcp"
cidr_blocks = ["0.0.0.0/0"]
}
# outbound internet access
egress {
from_port = 0
to_port = 0
main
0 0
0
AWS: profile:dev-user
Ln 102, Col 31
Spaces: 2
UTF-8
CRLF
Terraform
Prettier



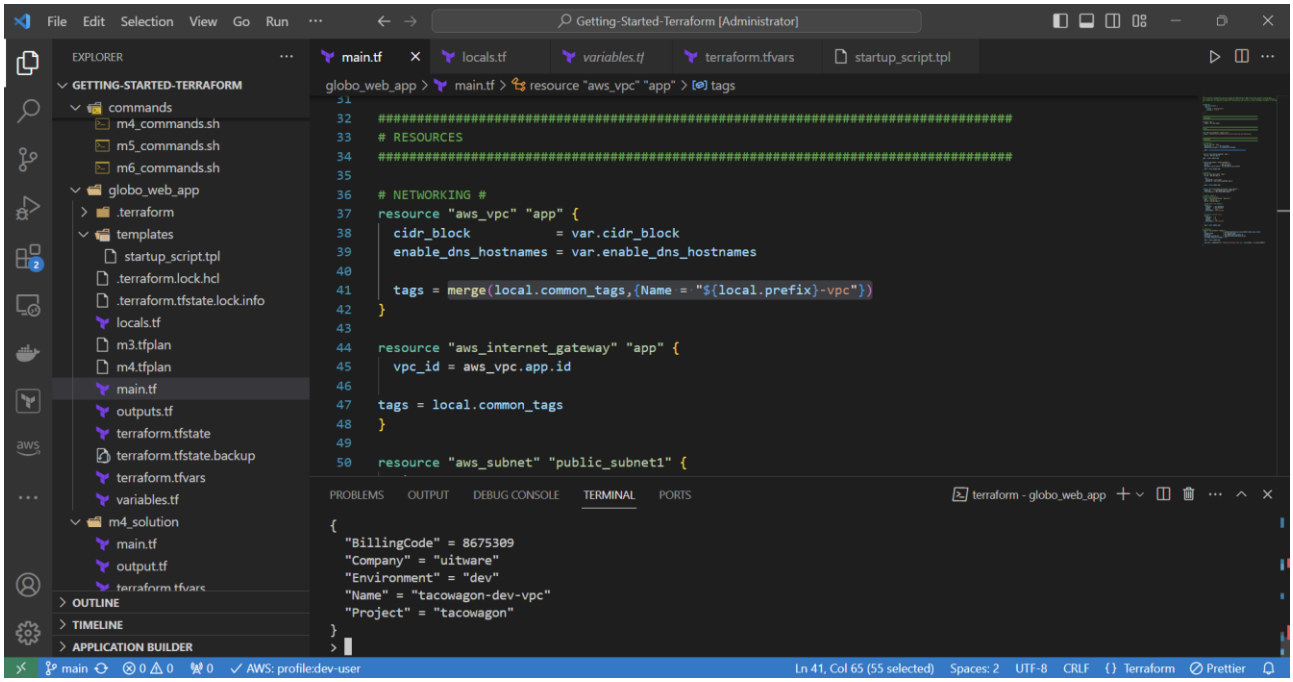
Testing terraform console



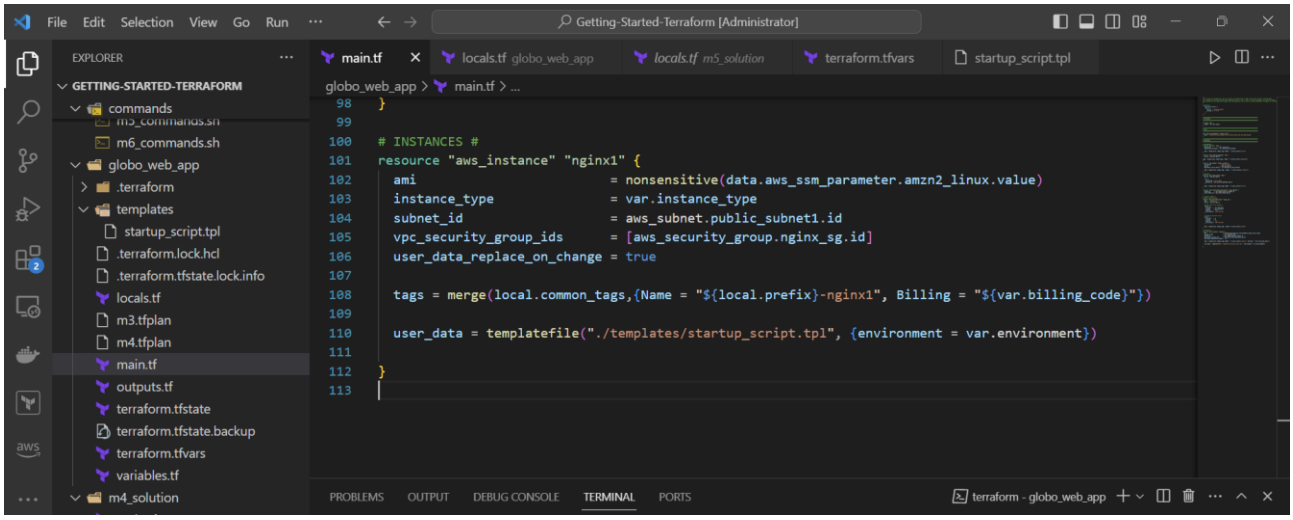
Adding a file startup_script.tpl with interpolation $(environment)

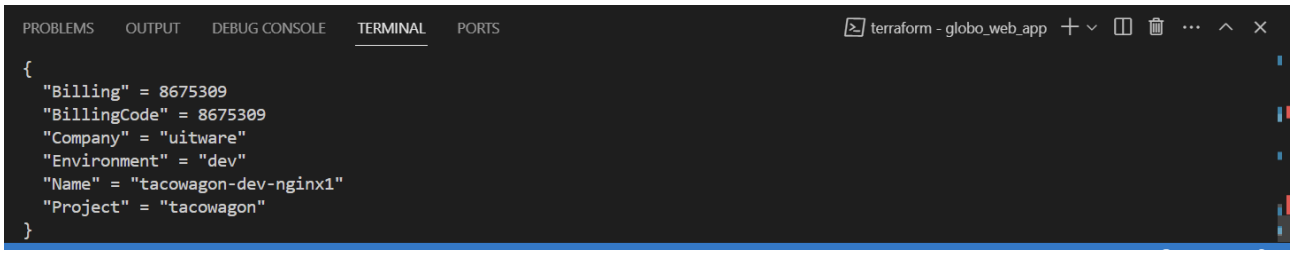


Also added function merge to tags and tested in terraform console

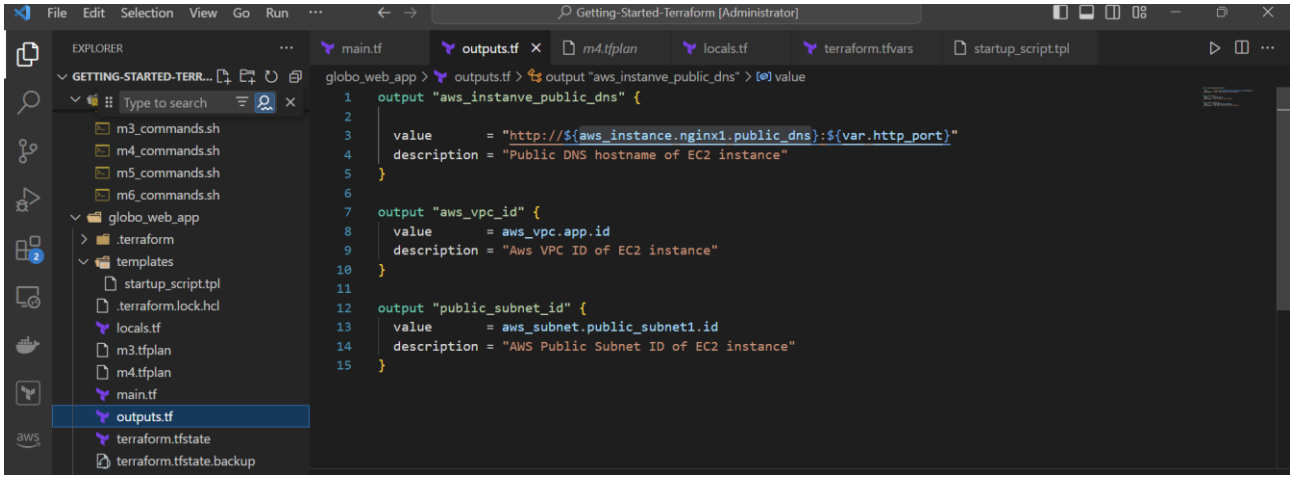


Also added custom merge function with billing code in map





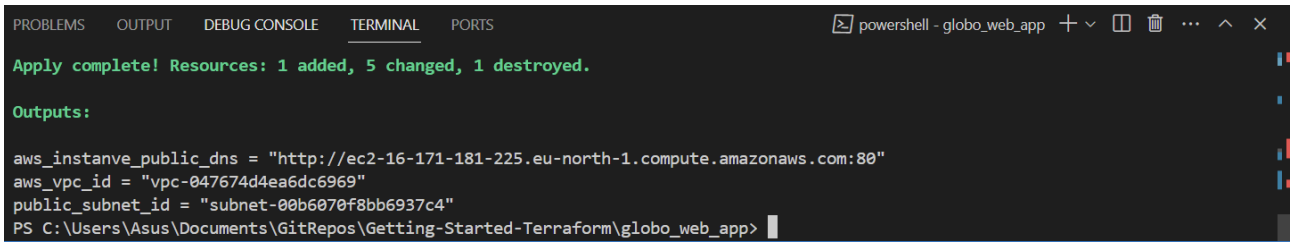
Also added clickable DNS output



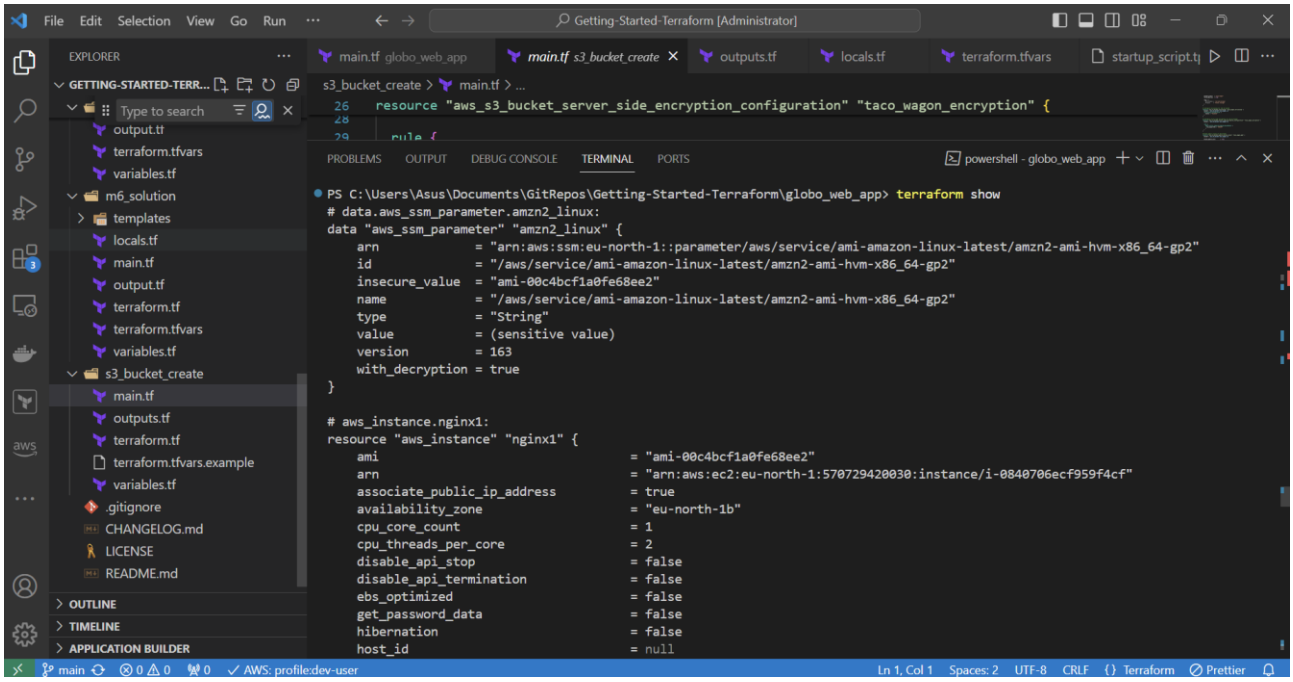
Formatting

```
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\globo_web_app> terraform fmt
locals.tf
main.tf
terraform.tfvars
variables.tf
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\globo_web_app>
```

Applied with a clickable dns



Using terraform state inspection commands




```
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\globo_web_app> terraform state list
data.aws_ssm_parameter.amzn2_linux
aws_instance.nginx1
aws_internet_gateway.app
aws_route_table.app
aws_route_table_association.app_subnet1
aws_security_group.nginx_sg
aws_subnet.public_subnet1
aws_vpc.app
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\globo_web_app>
```

```
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\globo_web_app> terraform state show aws_instance.nginx1
# aws_instance.nginx1:
resource "aws_instance" "nginx1" {
    ami                                  = "ami-00c4bcf1a0fe68ee2"
    arn                                  = "arn:aws:ec2:eu-north-1:570729420030:instance/i-0840706ecf959f4cf"
    associate_public_ip_address          = true
    availability_zone                    = "eu-north-1b"
    cpu_core_count                       = 1
    cpu_threads_per_core                 = 2
    disable_api_stop                     = false
    disable_api_termination              = false
    ebs_optimized                        = false
    get_password_data                    = false
    hibernation                          = false
    host_id                              = null
    iam_instance_profile                 = null
    id                                   = "i-0840706ecf959f4cf"
    instance_initiated_shutdown_behavior = "stop"
    instance_lifecycle                   = null
    instance_state                       = "running"
    instance_type                        = "t3.micro"
```

```
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\globo_web_app> terraform output
aws_instanve_public_dns = "http://ec2-16-171-181-225.eu-north-1.compute.amazonaws.com:80"
aws_vpc_id = "vpc-047674d4ea6dc6969"
public_subnet_id = "subnet-00b6070f8bb6937c4"
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\globo_web_app>
```

Creating s3 bucket

```
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\s3_bucket_create> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Installing hashicorp/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (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.
```

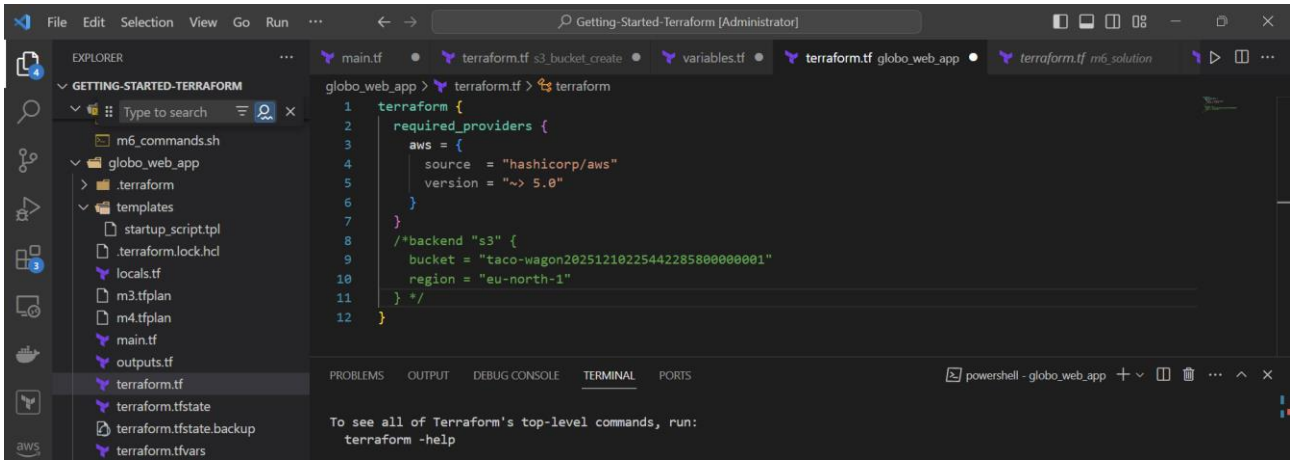
Creating plan

```
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\s3_bucket_create> terraform plan -out s3.tfplan

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_s3_bucket.taco_wagon will be created
  + resource "aws_s3_bucket" "taco_wagon" {
      + acceleration_status         = (known after apply)
      + acl                         = (known after apply)
      + arn                         = (known after apply)
      + bucket                      = (known after apply)
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = "taco-wagon"
      + bucket_regional_domain_name = (known after apply)
      + force_destroy               = true
      + 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                        = {
          + "Environment" = "terraform-demo"
          + "Purpose"     = "State storage"
        }
      + tags_all                    = {
          + "Environment" = "terraform-demo"
          + "Purpose"     = "State storage"
```

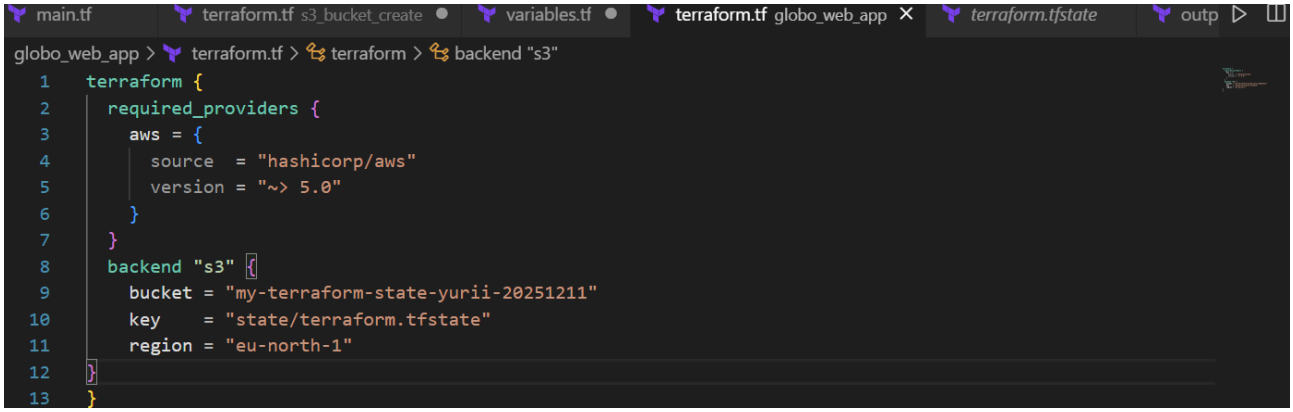
Applying

```
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\s3_bucket_create> terraform apply ./s3.tfplan
aws_s3_bucket.taco_wagon: Creating...
aws_s3_bucket.taco_wagon: Creation complete after 3s [id=taco-wagon20251210225442285800000001]
aws_s3_bucket_public_access_block.taco_wagon_pab: Creating...
aws_s3_bucket_server_side_encryption_configuration.taco_wagon_encryption: Creating...
aws_s3_bucket_versioning.taco_wagon_versioning: Creating...
aws_s3_bucket_public_access_block.taco_wagon_pab: Creation complete after 1s [id=taco-wagon20251210225442285800000001]
aws_s3_bucket_server_side_encryption_configuration.taco_wagon_encryption: Creation complete after 1s [id=taco-wagon202512102254
42285800000001]
aws_s3_bucket_versioning.taco_wagon_versioning: Creation complete after 2s [id=taco-wagon20251210225442285800000001]

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

Outputs:

bucket_name = "taco-wagon20251210225442285800000001"
bucket_region = "us-east-1"
```

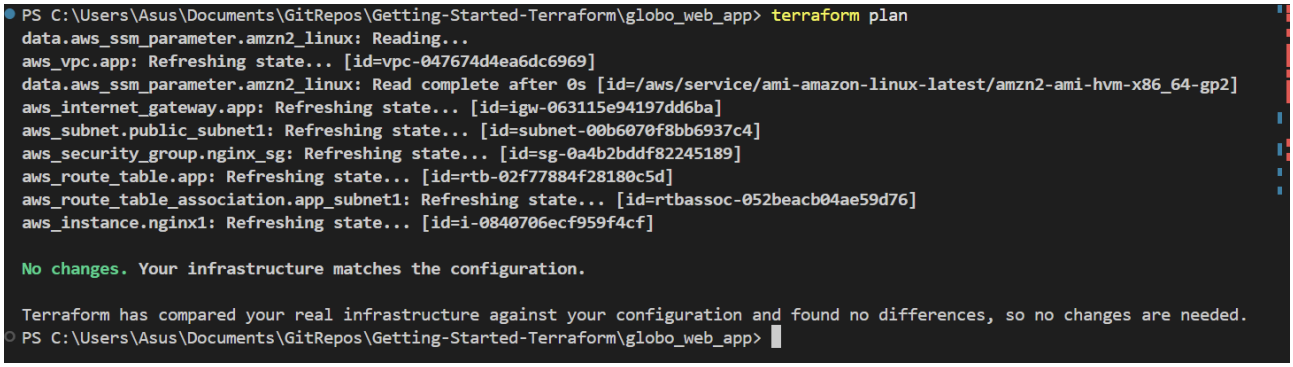
Creating s3 bucket for saving tfstate in cloud



Reconfiguring bucket for region eu-north-1



Creating plan



Terraform access to state data is fine

Running terraform destroy

```
PS C:\Users\Asus\Documents\GitRepos\Getting-Started-Terraform\globo_web_app> terraform destroy
data.aws_ssm_parameter.amzn2_linux: Reading...
aws_vpc.app: Refreshing state... [id=vpc-047674d4ea6dc6969]
data.aws_ssm_parameter.amzn2_linux: Read complete after 1s [id=/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2]
aws_internet_gateway.app: Refreshing state... [id=igw-063115e94197dd6ba]
aws_subnet.public_subnet1: Refreshing state... [id=subnet-00b6070f8bb6937c4]
aws_security_group.nginx_sg: Refreshing state... [id=sg-0a4b2bddf82245189]
aws_route_table.app: Refreshing state... [id=rtb-02f77884f28180c5d]
aws_instance.nginx1: Refreshing state... [id=i-0840706ecf959f4cf]
aws_route_table_association.app_subnet1: Refreshing state... [id=rtbassoc-052beacb04ae59d76]

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_instance.nginx1 will be destroyed
  - resource "aws_instance" "nginx1" {
      - ami                                  = "ami-00c4bcf1a0fe68ee2" -> null
      - arn                                  = "arn:aws:ec2:eu-north-1:570729420030:instance/i-0840706ecf959f4cf" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "eu-north-1b" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 2 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
```

```
aws_route_table.app: Destroying... [id=rtb-02f77884f28180c5d]
aws_route_table.app: Destruction complete after 1s
aws_internet_gateway.app: Destroying... [id=igw-063115e94197dd6ba]
aws_instance.nginx1: Still destroying... [id=i-0840706ecf959f4cf, 00m10s elapsed]
aws_internet_gateway.app: Still destroying... [id=igw-063115e94197dd6ba, 00m10s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0840706ecf959f4cf, 00m20s elapsed]
aws_internet_gateway.app: Still destroying... [id=igw-063115e94197dd6ba, 00m20s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0840706ecf959f4cf, 00m30s elapsed]
aws_internet_gateway.app: Still destroying... [id=igw-063115e94197dd6ba, 00m30s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0840706ecf959f4cf, 00m40s elapsed]
aws_internet_gateway.app: Still destroying... [id=igw-063115e94197dd6ba, 00m40s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0840706ecf959f4cf, 00m50s elapsed]
aws_internet_gateway.app: Still destroying... [id=igw-063115e94197dd6ba, 00m50s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0840706ecf959f4cf, 01m00s elapsed]
aws_internet_gateway.app: Still destroying... [id=igw-063115e94197dd6ba, 01m00s elapsed]
aws_internet_gateway.app: Destruction complete after 1m8s
aws_instance.nginx1: Still destroying... [id=i-0840706ecf959f4cf, 01m10s elapsed]
aws_instance.nginx1: Destruction complete after 1m12s
aws_subnet.public_subnet1: Destroying... [id=subnet-00b6070f8bb6937c4]
aws_security_group.nginx_sg: Destroying... [id=sg-0a4b2bddf82245189]
aws_subnet.public_subnet1: Destruction complete after 0s
aws_security_group.nginx_sg: Destruction complete after 0s
aws_vpc.app: Destroying... [id=vpc-047674d4ea6dc6969]
aws_vpc.app: Destruction complete after 1s

Destroy complete! Resources: 7 destroyed.
```