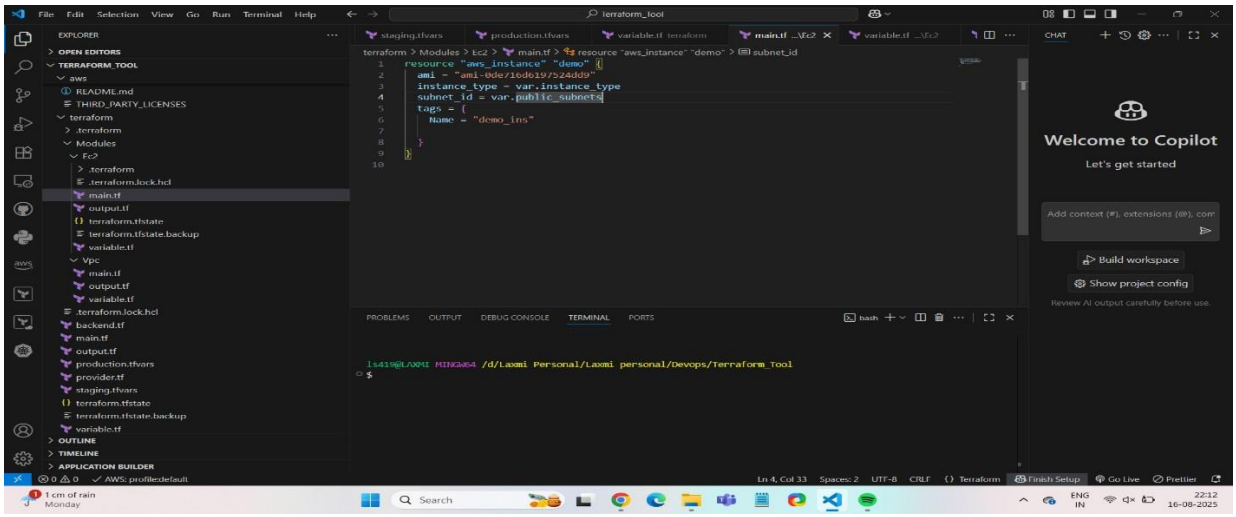
Name :- Laxmi Swami

Topic :- Vpc Connect to Instance using Terraform

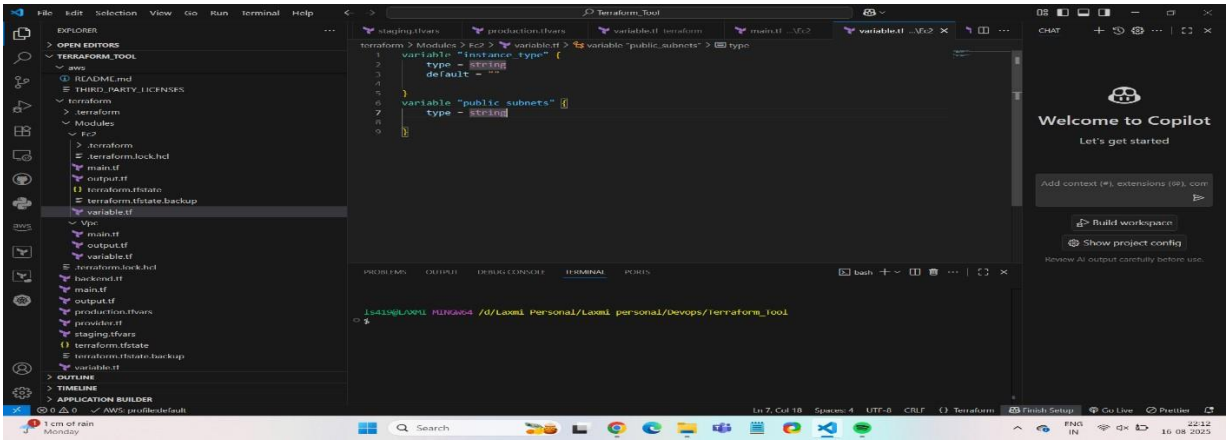
STEP 1 :- If you want to create an ec2 instance using same vpc which you going to create go to ec2>main.tf type **subnet_id = var.public_subnets**



STEP 2 :- Now as we have defined some variable in main.tf we have to define it in ec2>variable.tf folder too. So type below script **variable "public_subnets" {**

Type = string

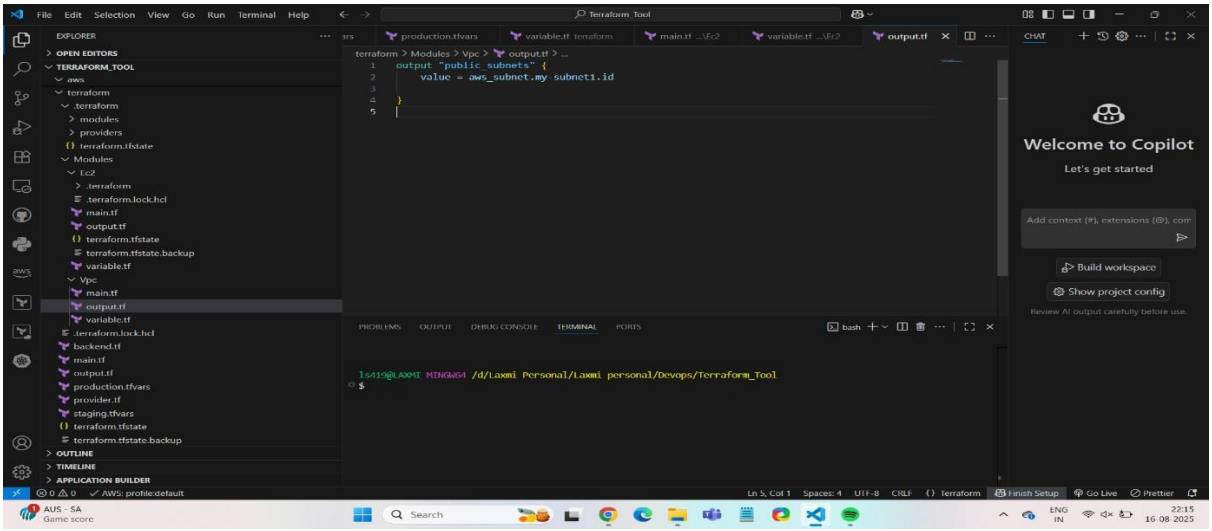
}



STEP 3 :- Now go to vpc >output.tf and mention the subnet value which we need to attach to our instance. **output "public_subnets" {**

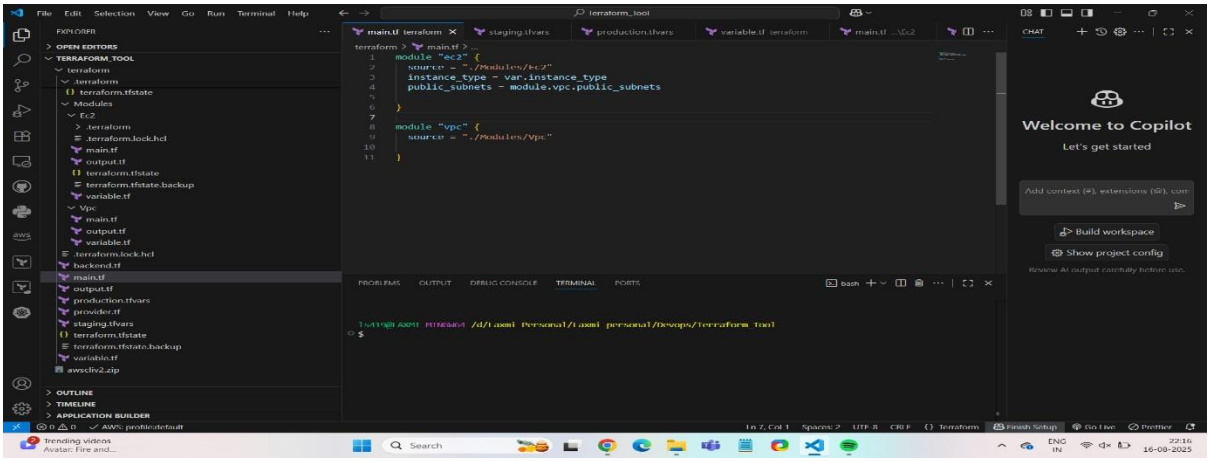
value = aws_subnet.my-subnet1.id

}

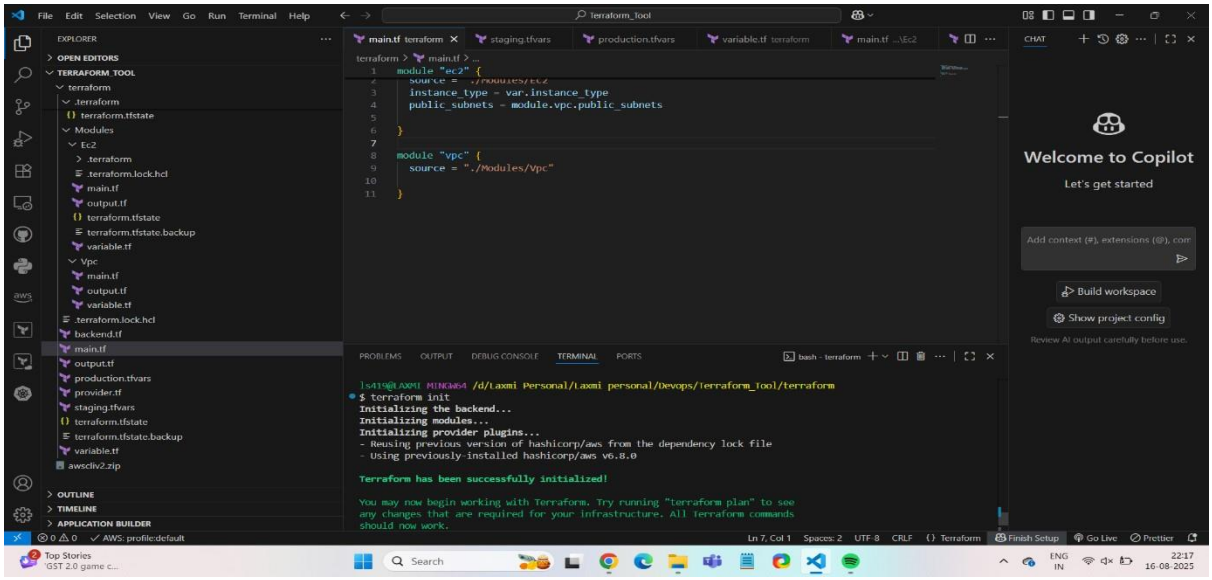


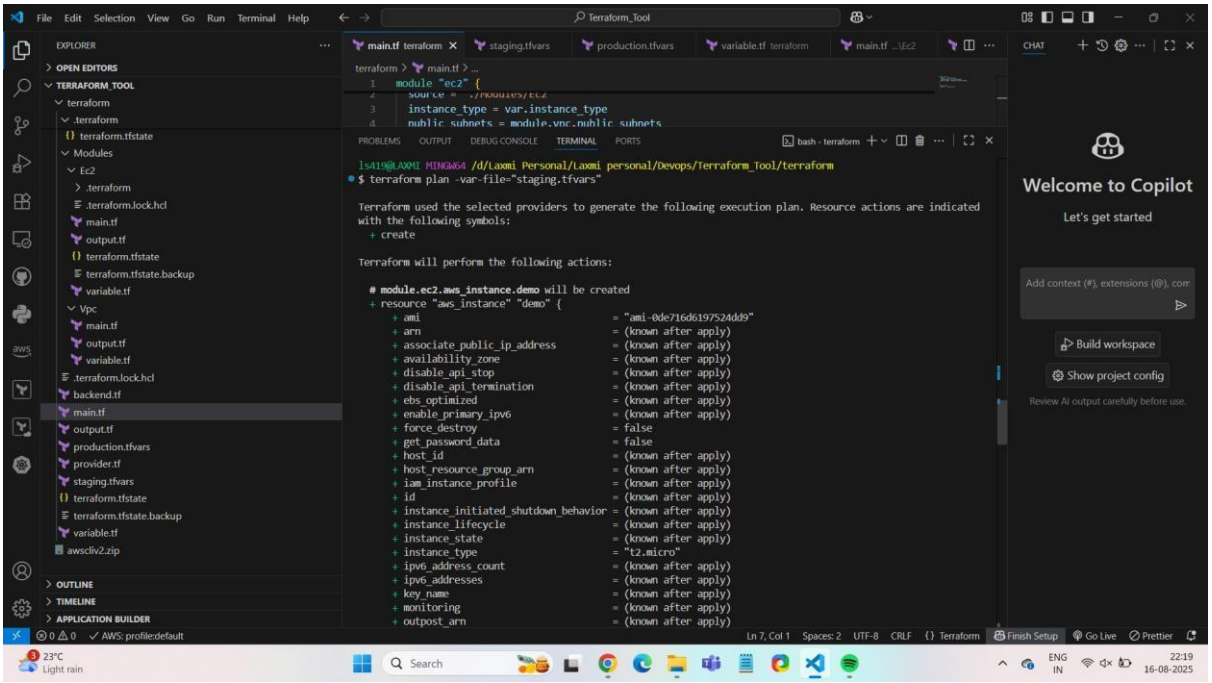
STEP 4 :- Now go to terraform>main.tf and mention subnet type

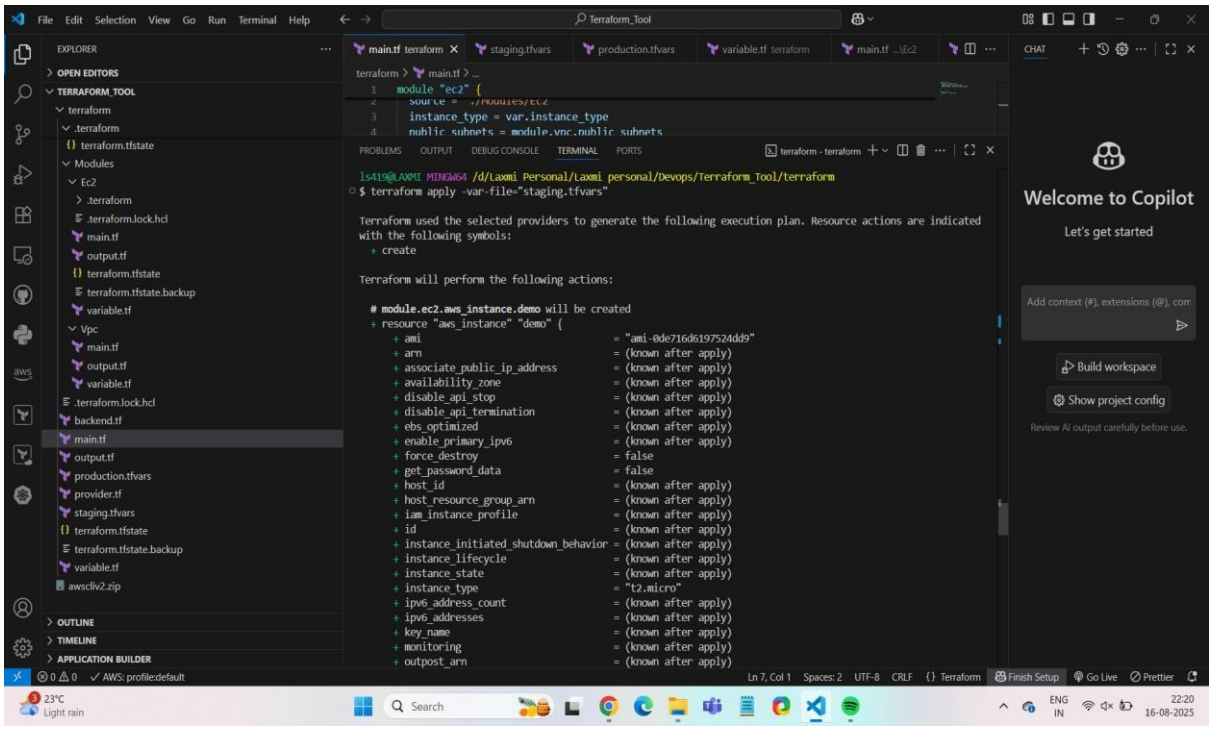
public_subnets =module.vpc.public_subnets



STEP 5 :- Now go to terraform folder and fetch below commands **terraform init terraform plan -var-file.="staging.tfvars" terraform apply -var-file.="staging.tfvars"**

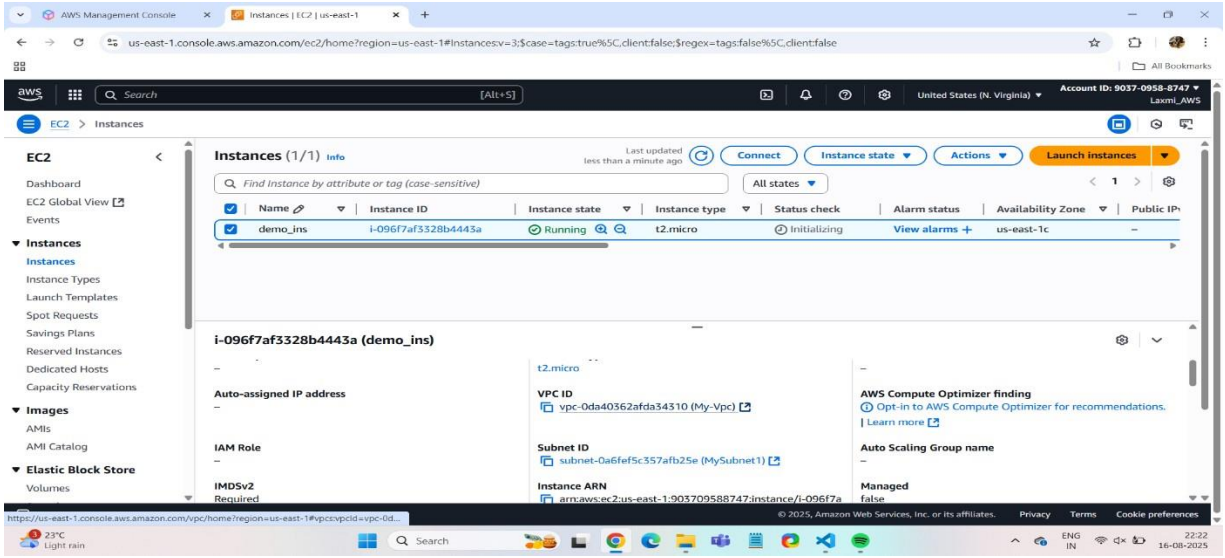

File Edit Selection View Go Run Terminal Help
terraform_tool
EXPLORER
OPEN EDITORS
TERRAFORM_TOOL
terraform
.terraform
terraform.tfstate
Modules
Ec2
.terraform
.terraform.lock.hcl
main.tf
output.tf
terraform.tfstate
terraform.tfstate.backup
variable.tf
Vpc
main.tf
output.tf
variable.tf
.terraform.lock.hcl
backend.tf
main.tf
output.tf
production.tfvars
provider.tf
staging.tfvars
terraform.tfstate
terraform.tfstate.backup
variable.tf
awscliv2.zip
OUTLINE
TIMELINE
APPLICATION BUILDER
main.tf terraform
staging.tfvars
production.tfvars
variable.tf terraform
main.tf ...\Ec2
terraform > main.tf > ...
module "ec2" {
source = "./Modules/Ec2"
instance_type = var.instance_type
public_subnets = module.vpc.public_subnets
}
module "vpc" {
source = "./Modules/Vpc"
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash - terraform
1s419@LAXMI MINGW64 /d/Laxmi Personal/Laxmi personal/Devops/Terraform_Tool/terraform
$ terraform init
Initializing the backend...
Initializing modules...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.8.0
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.
CHAT
Welcome to Copilot
Let's get started
Build workspace
Show project config
Review AI output carefully before use.
AWS: profile:default
Ln 7, Col 1
Spaces: 2
UTF-8
CRLF
Terraform
Finish Setup
Go Live
Prettier
Top Stories
GST 2.0 game c...
Search
ENG
IN
22:17
16-08-2025



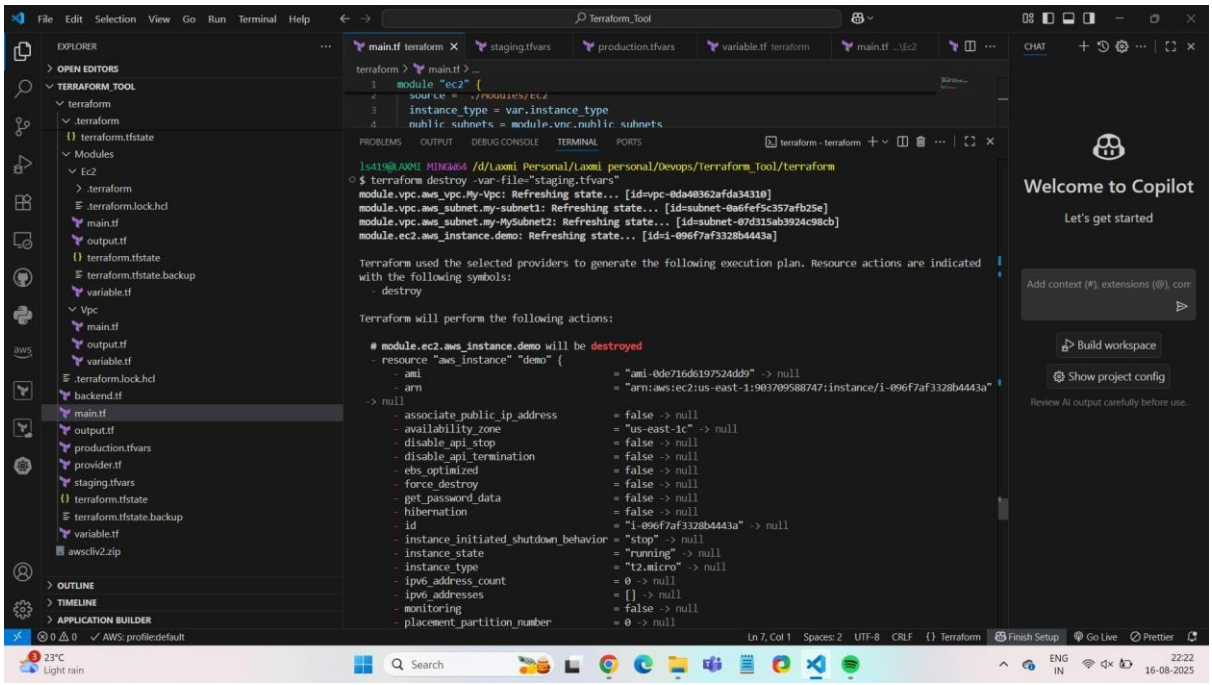


STEP 6 :- Now check the vpc and subnet created in instance type. For that go to networking and see the vpc and subnet same as we mention in our code. Obviously inside s3 bucket terraform-state file will also be modified.

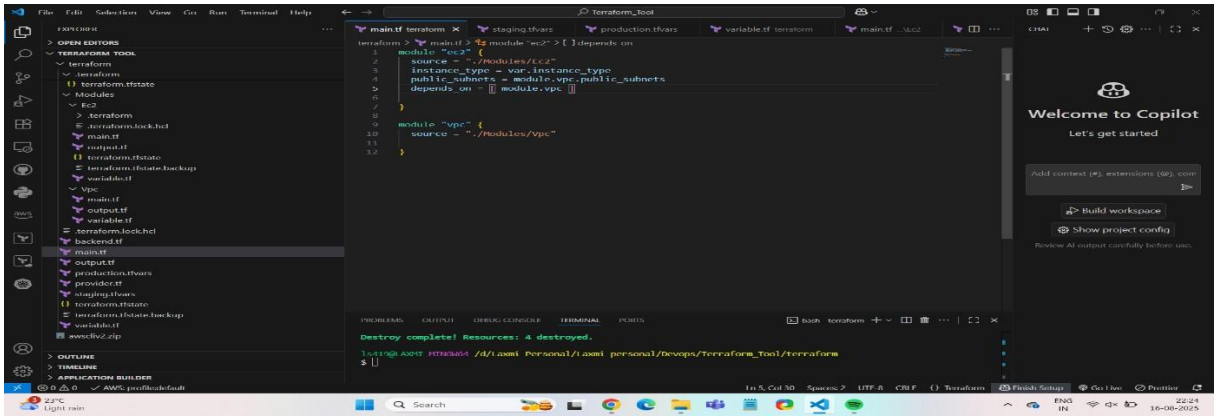
*Note first vpc then subnet and at the end instance will be created.



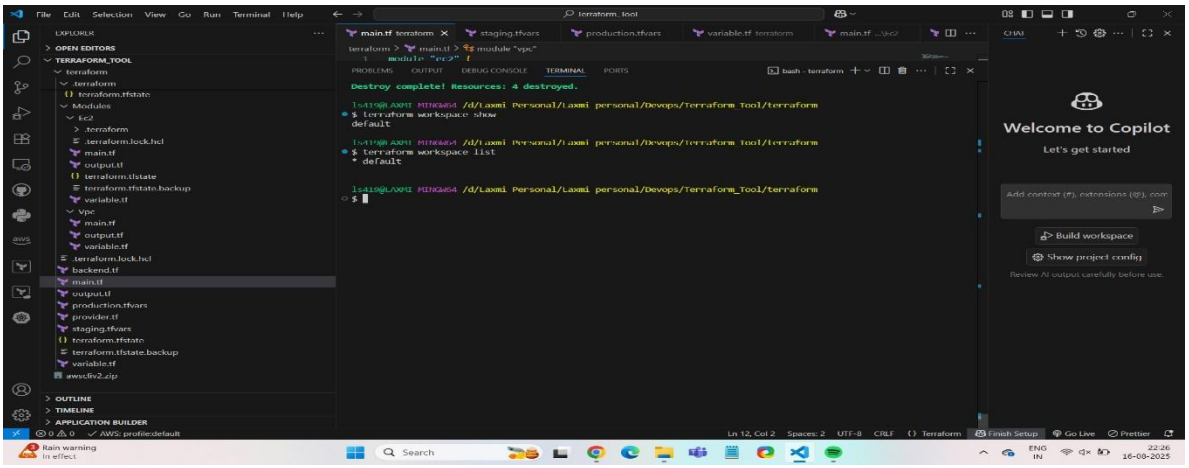
STEP 7 :- Now fetch **terraform destroy -varfile="staging.tfvars"** to destroy everything inside staging environment.

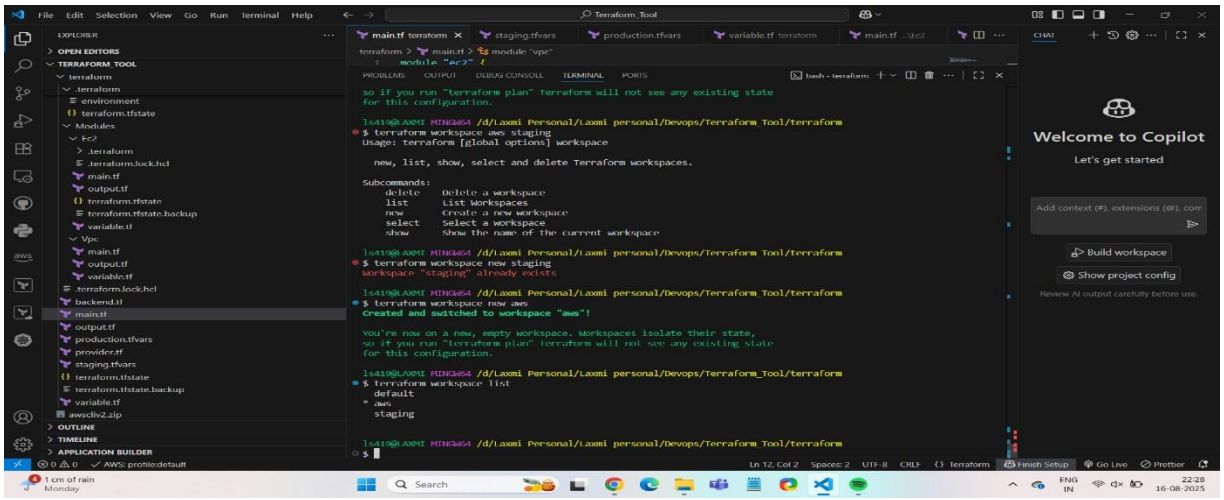


STEP 8 :- Now go to ec2 main.tf and add one more **depends_on = [module.vpc]** In this way also we can create instance using same vpc and subnet as we mention in our code.

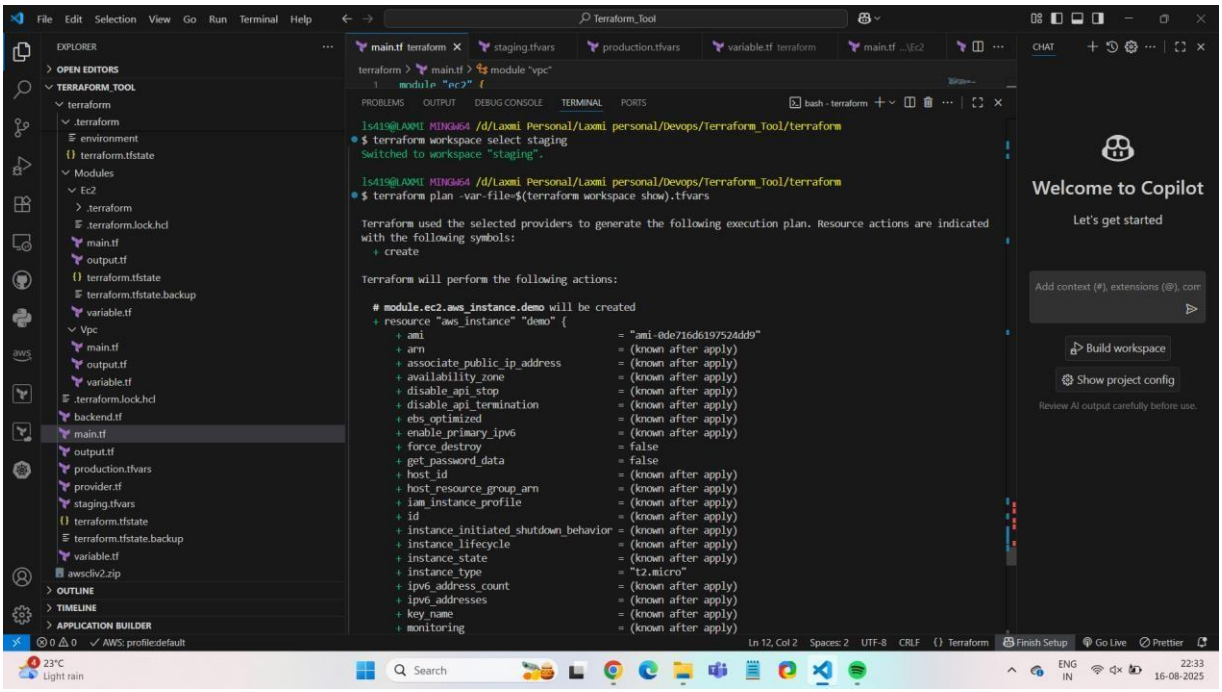


STEP 9 :- Now if we have to list create+switch and switch the workplace fetch below commands **terraform workspace show** (To show the current workspace) **terraform workspace list** (To list out the workspace) **terraform workspace staging** (To create a staging workspace also you will switch to it) **terraform select staging** (To switch to staging workspace)

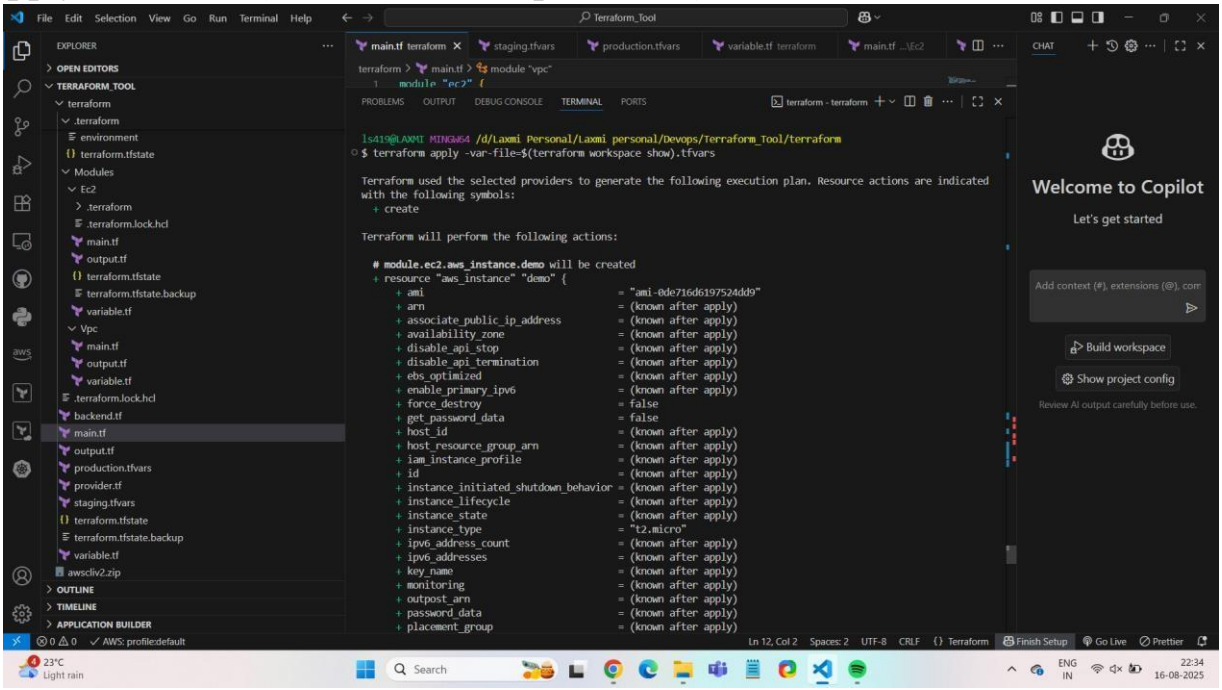


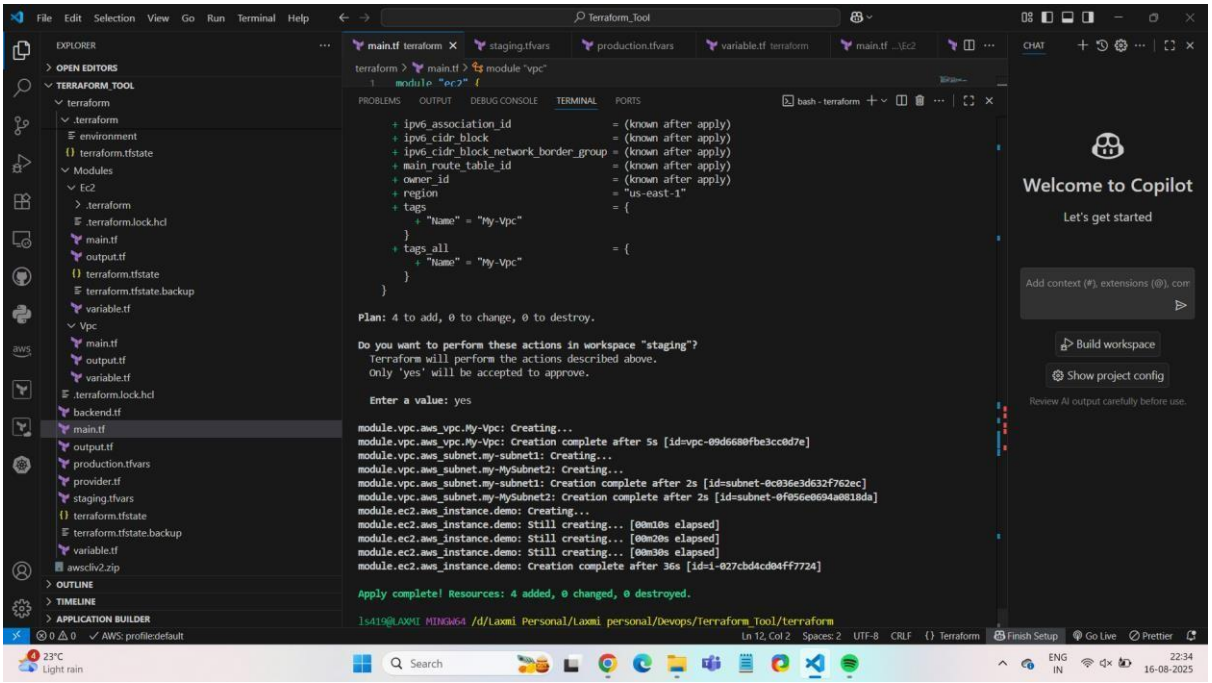


STEP 9 :-Now switch to staging workspace and fetch below command **terraform plan -var-file=$(terraform workspace show).tfvars**

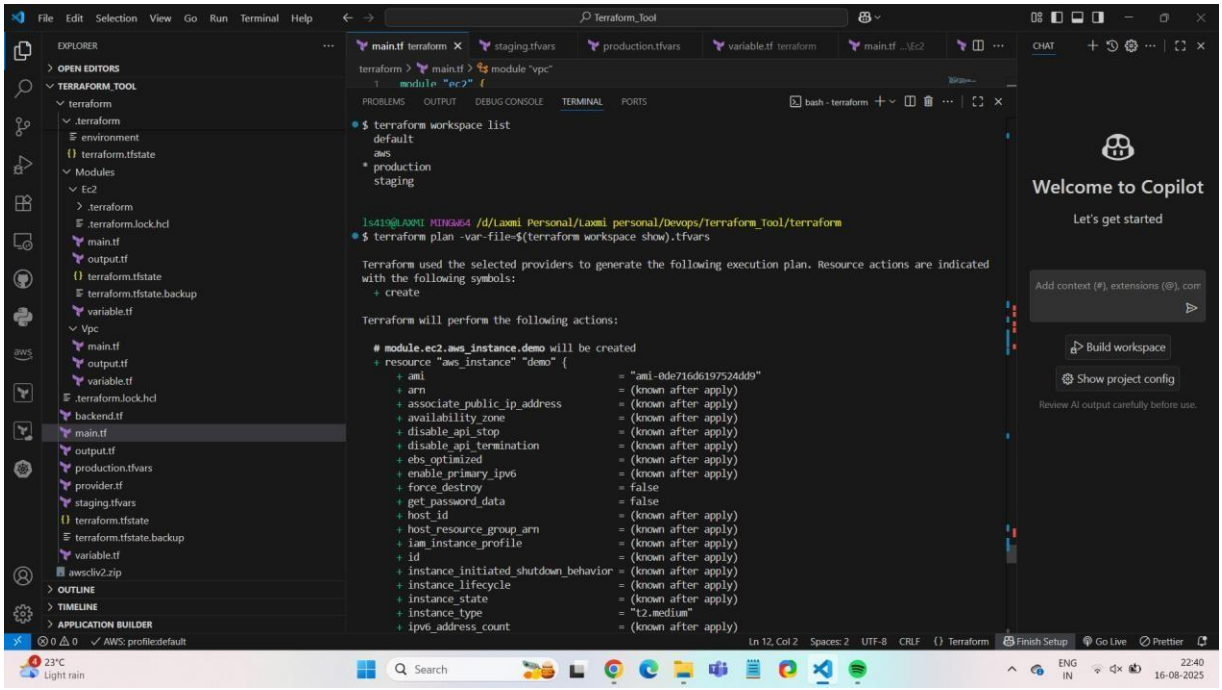


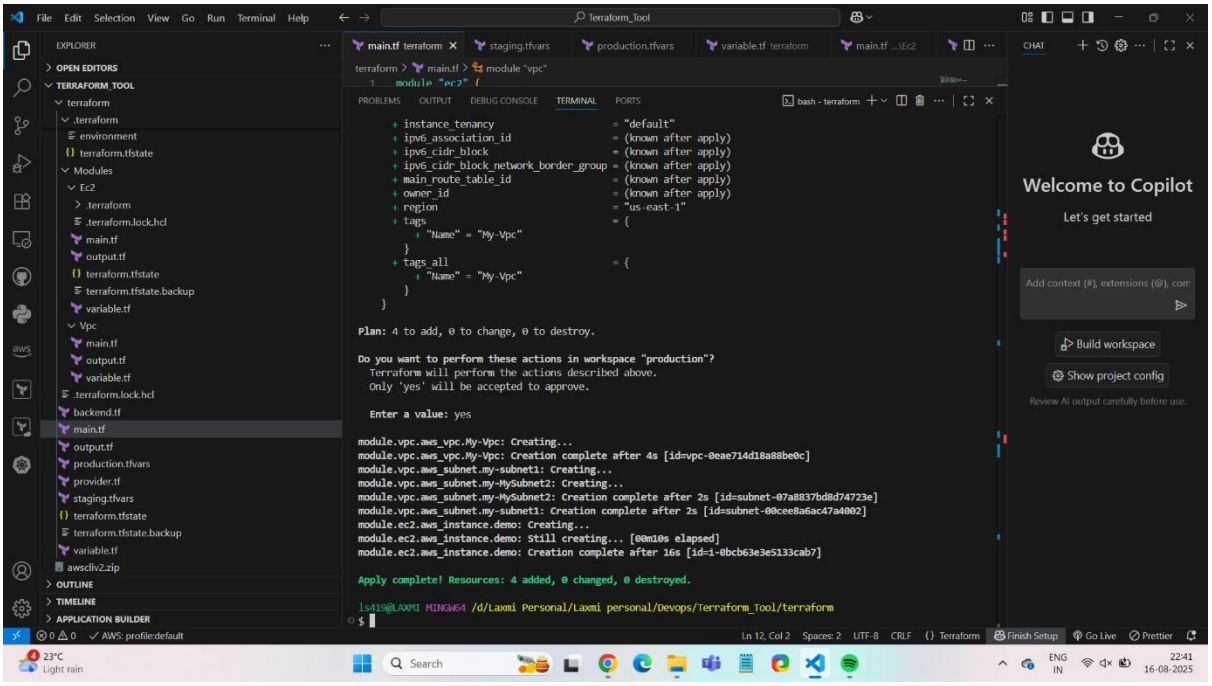
STEP 7 :- Now fetch below command to apply the changes **terraform apply -var-file=$(terraform workspace show).tfvars**



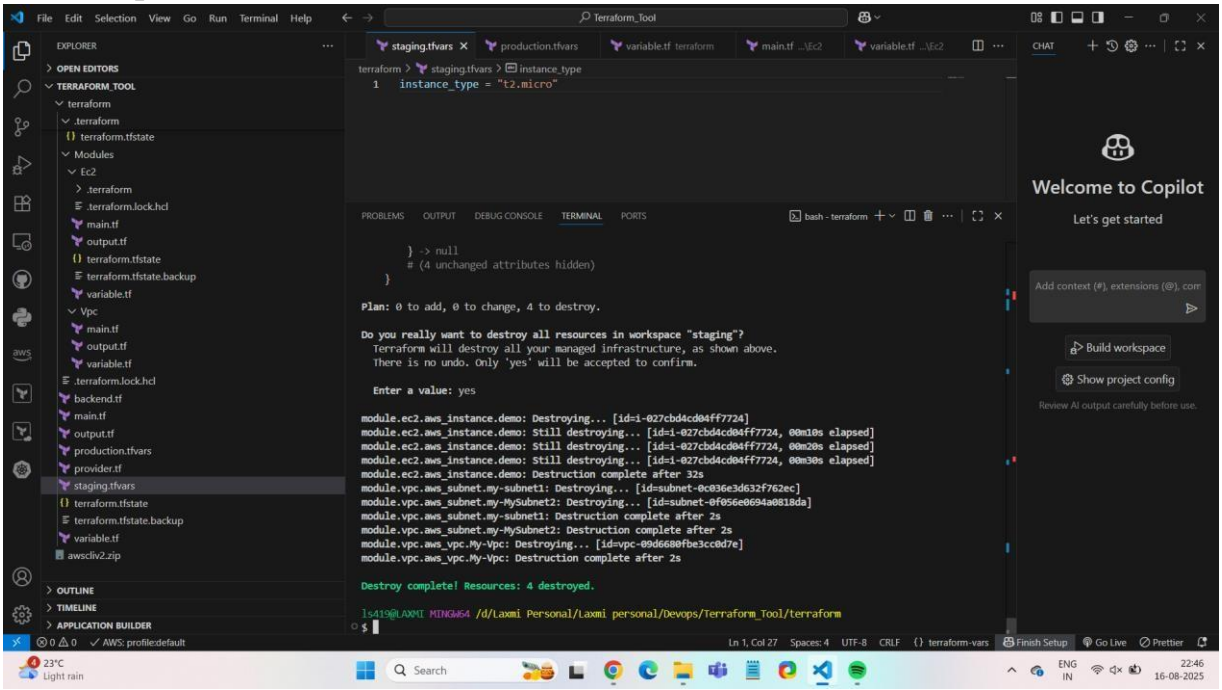


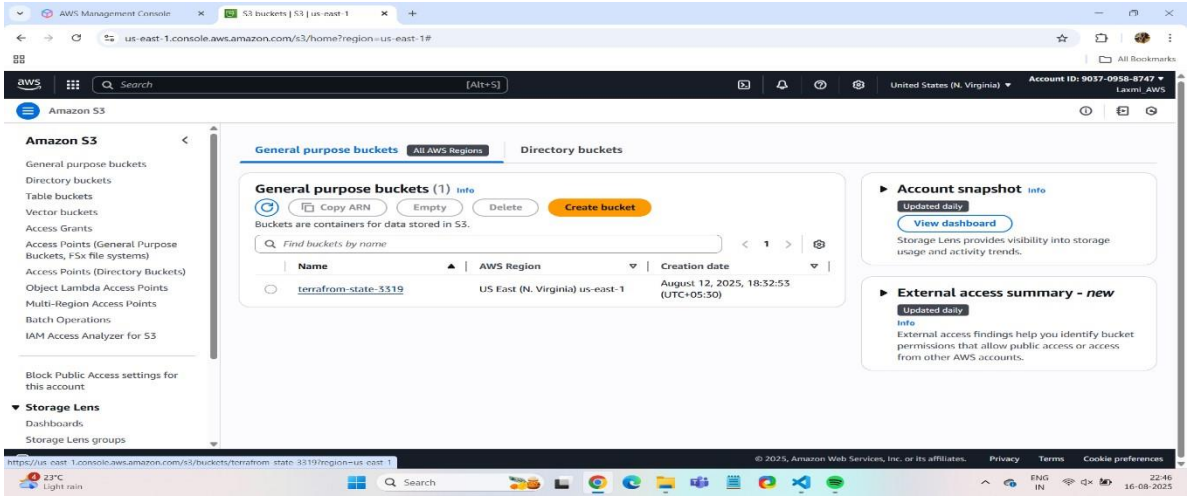
STEP11:Now switch to production workspace and fetch the plan and apply command.

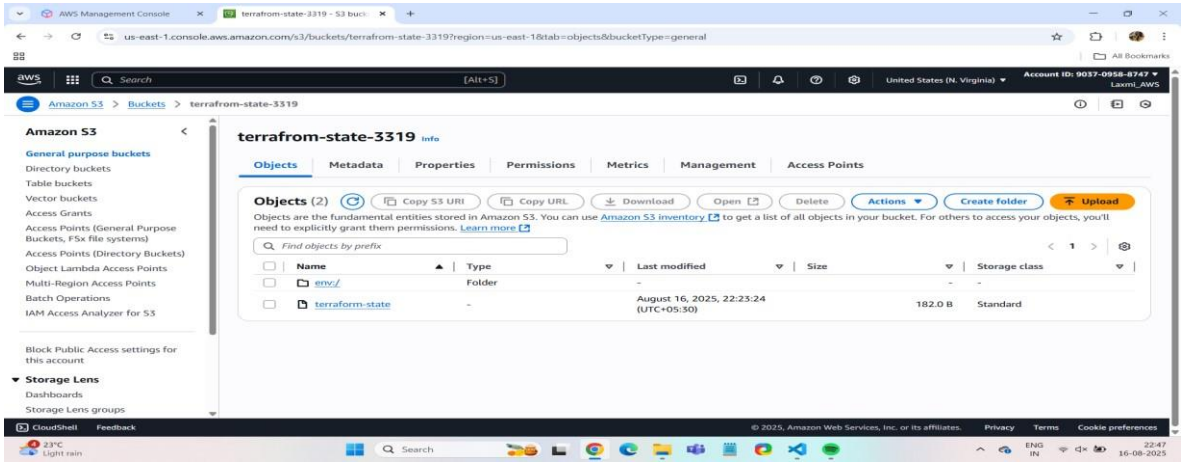


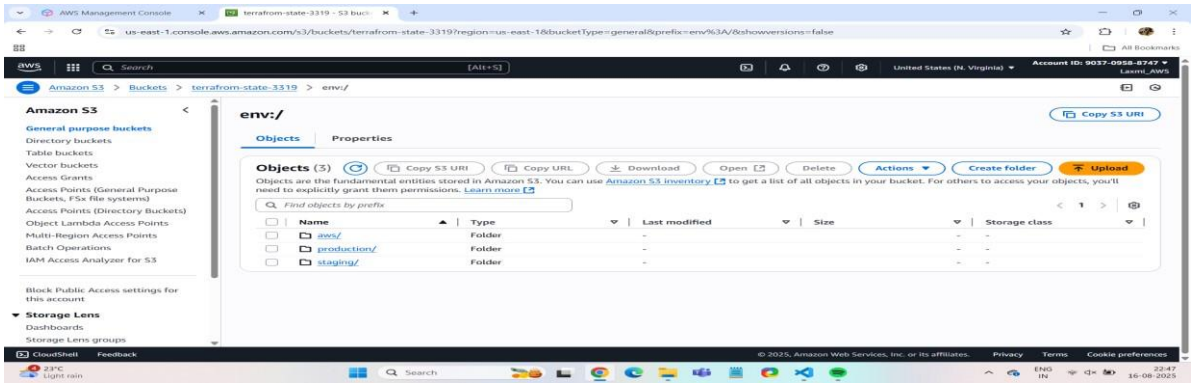


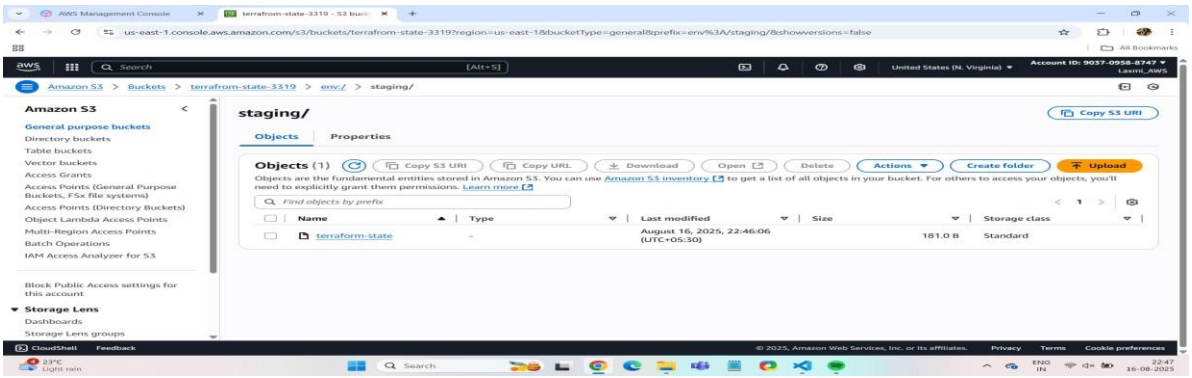
STEP12:It is obvious that 2 separate instance 2 separate state files inside s3 bucket will be created so destroy the task done by switching in both the workspace.



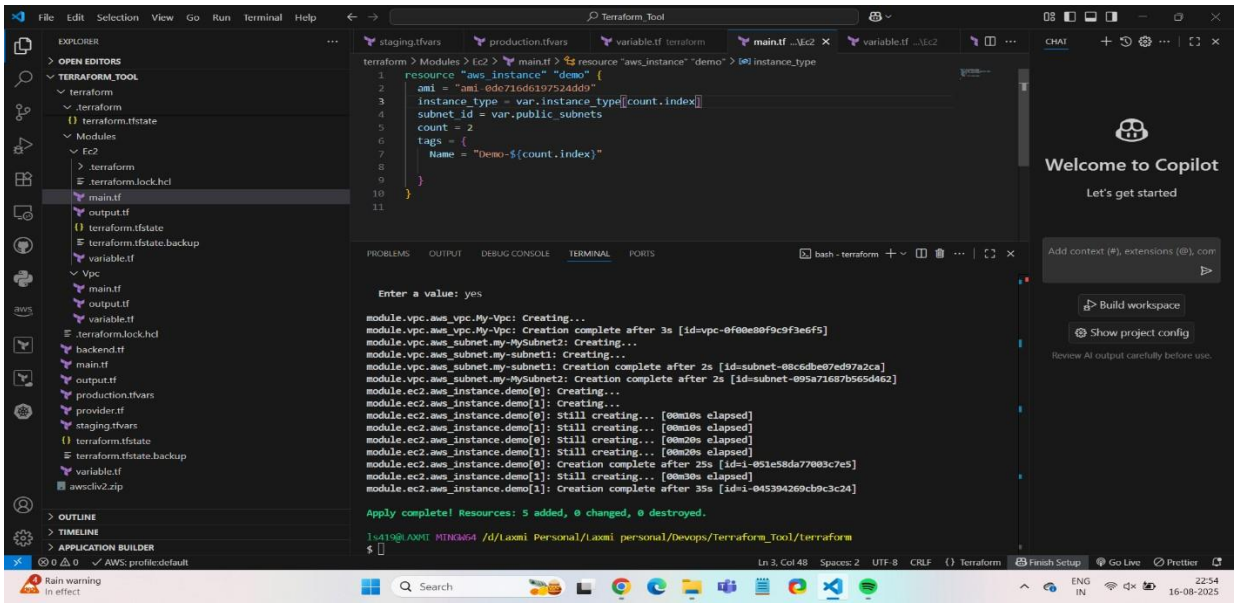

AWS Management Console
S3 buckets | S3 | us-east-1
us-east-1.console.aws.amazon.com/s3/home?region=us-east-1#
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
General purpose buckets (1)
Copy ARN
Empty
Delete
Create bucket
Buckets are containers for data stored in S3.
Find buckets by name
Name
AWS Region
Creation date
terrafrom-state-3319
US East (N. Virginia) us-east-1
August 12, 2025, 18:32:53 (UTC+05:30)
Account snapshot
View dashboard
Storage Lens provides visibility into storage usage and activity trends.
External access summary - new
External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


us-east-1.console.aws.amazon.com/s3/buckets/terrafrom-state-3319?region=us-east-1&tab=objects&bucketType=general
Amazon S3 > Buckets > terrafrom-state-3319
terrafrom-state-3319
Objects
Metadata
Properties
Permissions
Metrics
Management
Access Points
Objects (2)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
env:/
Folder
terraform-state
August 16, 2025, 22:23:24 (UTC+05:30)
182.0 B
Standard

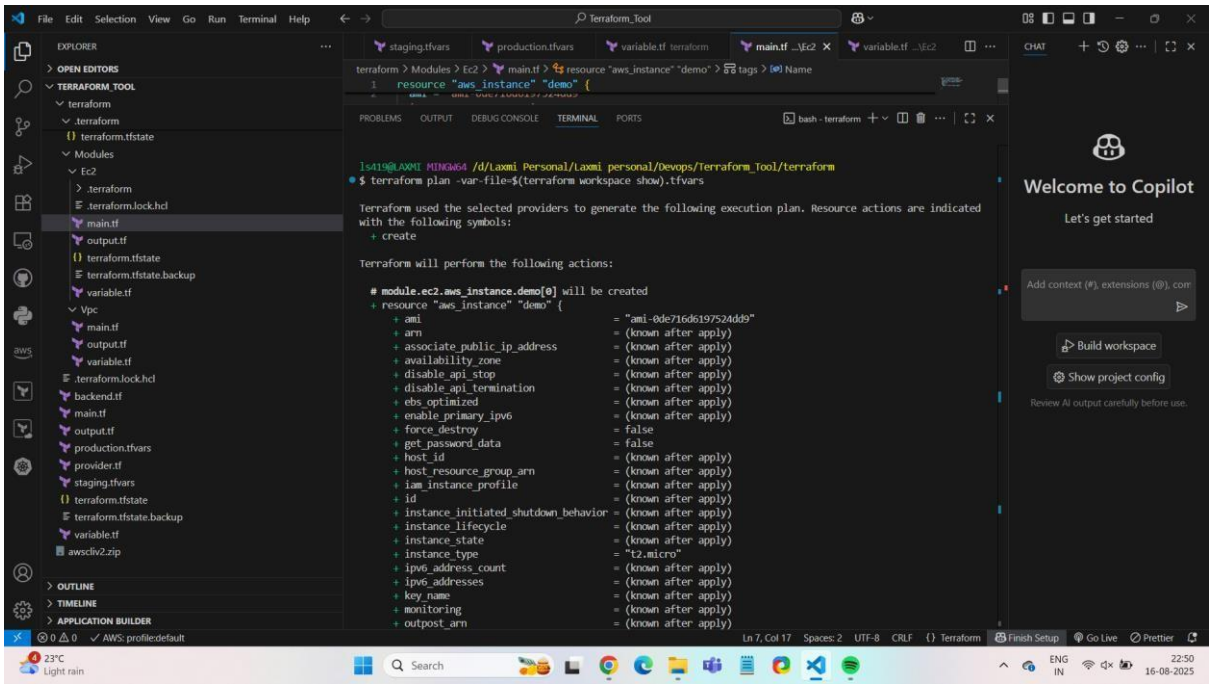

us-east-1.console.aws.amazon.com/s3/buckets/terrafrom-state-3319?region=us-east-1&bucketType=general&prefix=env%3A/&showversions=false
Amazon S3 > Buckets > terrafrom-state-3319 > env:/
env:/
Copy S3 URI
Objects
Properties
Objects (3)
aws/
production/
staging/

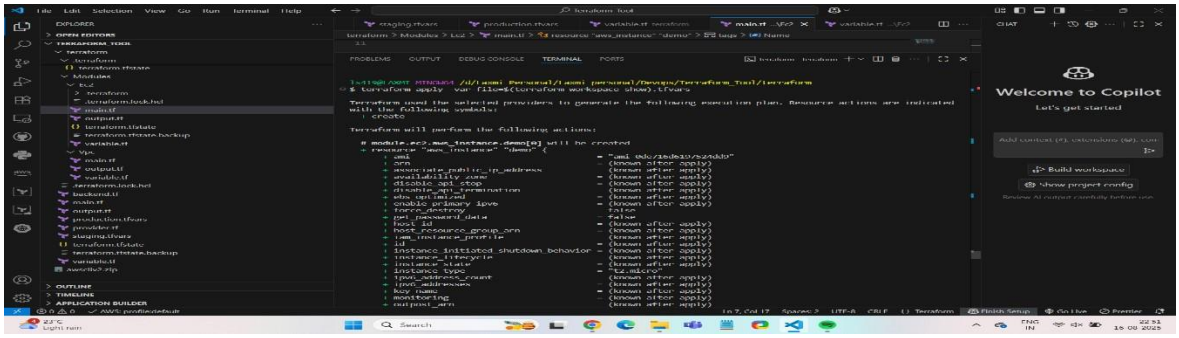

us-east-1.console.aws.amazon.com/s3/buckets/terrafrom-state-3319?region=us-east-1&bucketType=general&prefix=env%3A/staging/&showversions=false
Amazon S3 > Buckets > terrafrom-state-3319 > env:/ > staging/
staging/
Copy S3 URI
Objects
Properties
Objects (1)
terraform-state
August 16, 2025, 22:46:06 (UTC+05:30)
181.0 B
Standard

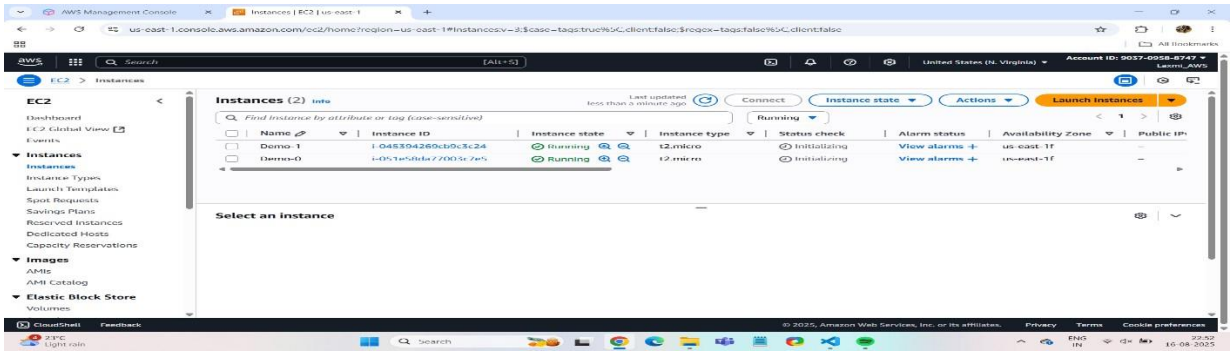
STEP 13 :- Now suppose we have to create 2 instances at a time modify the code inside ec2>main.tf and add below line in it **count = 2**



STEP 14 :- Now again switch to staging workspace and fetch plan and apply command.

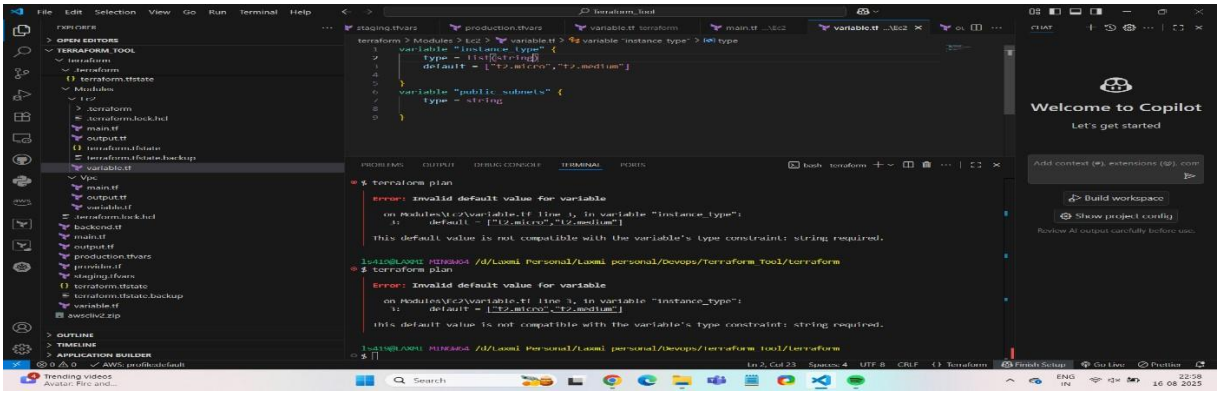






STEP 15 :- Now if we have to create 2 instance and give them different instance type then add below script in ec2 > variable.tf folder variable “instance_type**s**”

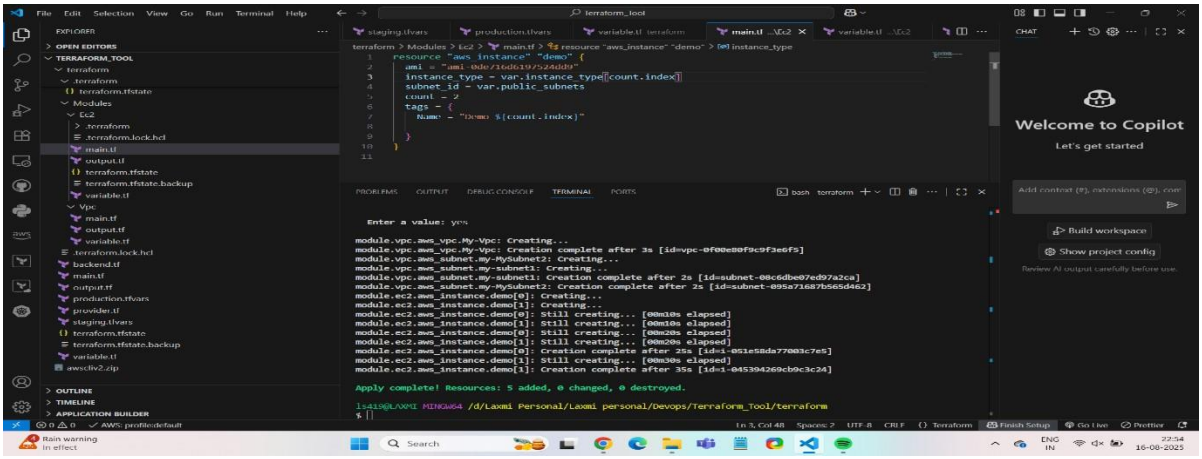
default = [“t2.micro”,”t2.medium”]



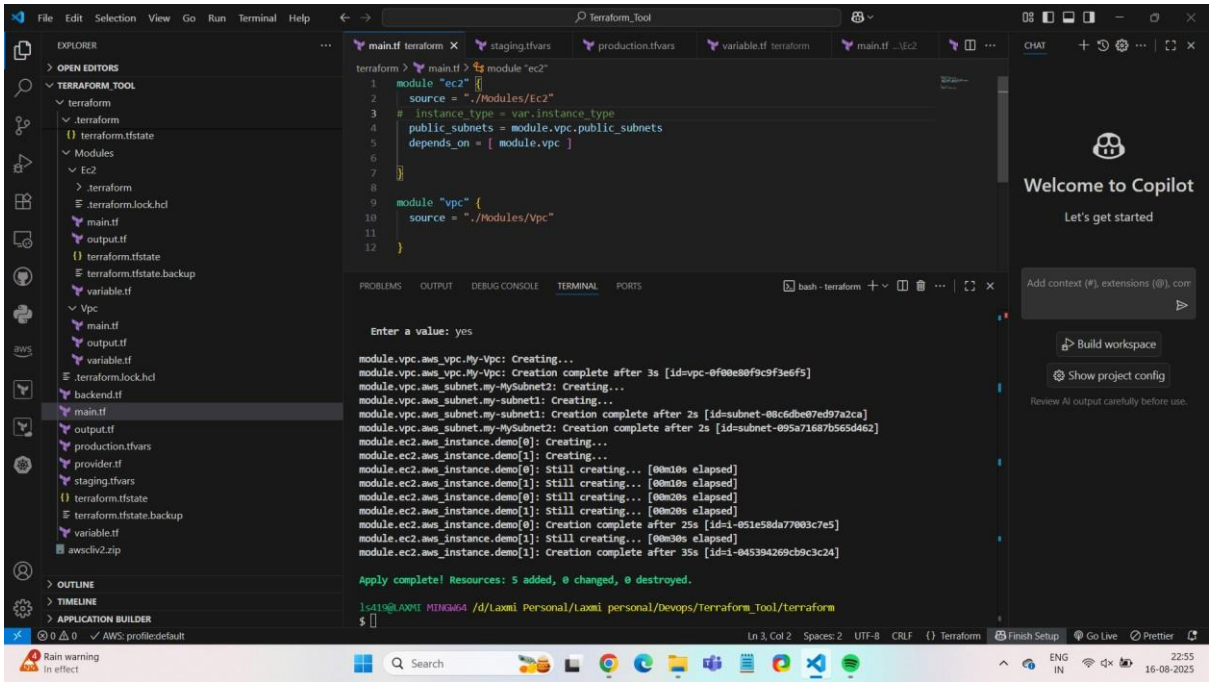
STEP 16 :- Now go to ec2>main.tf and modify script as below

Name= “Instance-$[count.index]

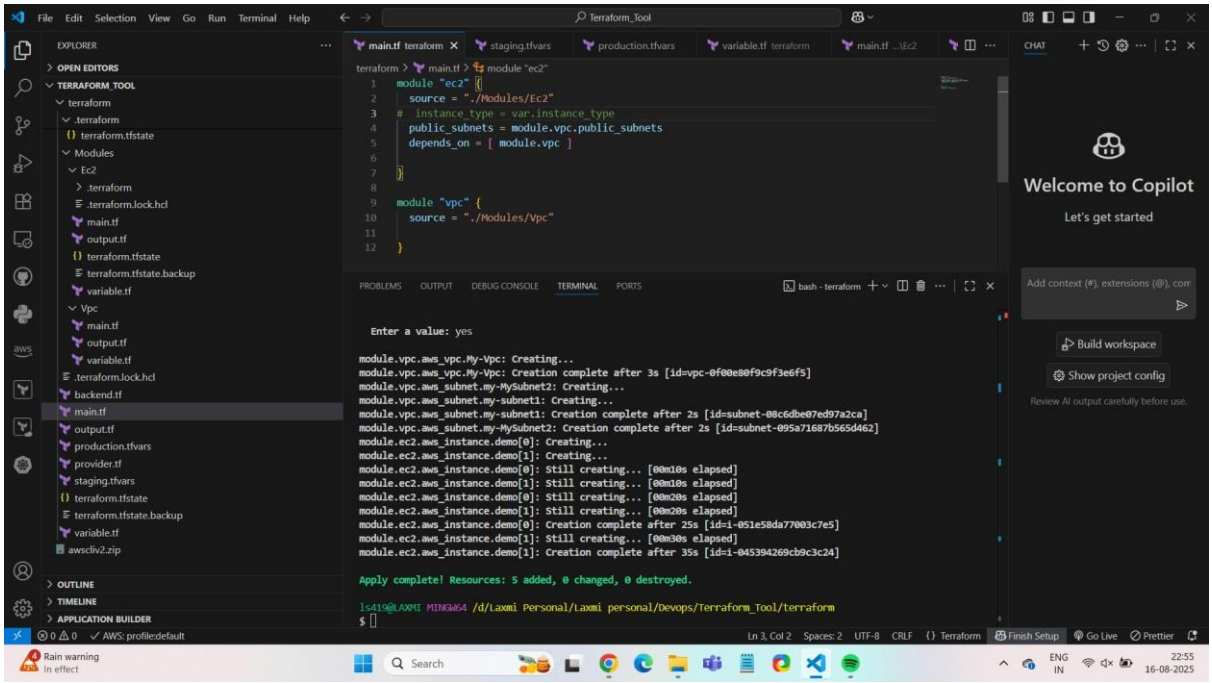
So that instance name will be different for each instance created.



STEP 17 :- Now go to terraform>main.tf and just comment put instance_type as we not going to use here.



STEP 18 :- Now plan and apply the code in staging environment.



STEP 19 :- Now destroy everything created as we have completed our task.