Task:2 **Himansh Sharma**
PearlThoughts
Learn AWS core concepts and Terraform **Date: 03/02/2026**
—--

AWS EC2 Manual & Terraform Provisioning

Objective

- Learn AWS core concepts
- Launch an EC2 instance manually using the AWS Console
- Provision an EC2 instance using Terraform
- Document the complete process

AWS Core Concepts

- **AWS Region**: Physical location where AWS data centers exist (e.g., ap-south-1)
- **Availability Zone (AZ)**: Isolated data centers within a region
- **EC2**: Virtual server in AWS
- **AMI**: Template used to create EC2 instances
- **Instance Type**: Defines CPU, memory, and network capacity
- **Key Pair**: Used for SSH access
- **Security Group**: Acts as a virtual firewall
- **IAM**: Manages access and permissions

Part 1: Launch EC2 Instance Manually (AWS Console)

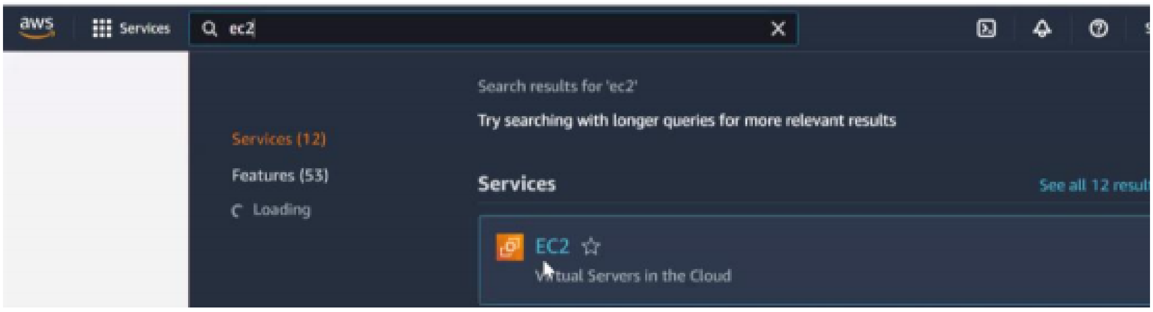
Steps

1. Login to **AWS Management Console**
2. Navigate to **EC2 → Instances → Launch Instance**
3. Choose an **AMI** (Amazon Linux)
4. Select **Instance Type** (t2.micro – Free Tier)
5. Create or select an **Key Pair**
6. Configure **Security Group**:
   - Allow SSH (22)
7. Review and **Launch Instance**

Task:2 **Himansh Sharma**

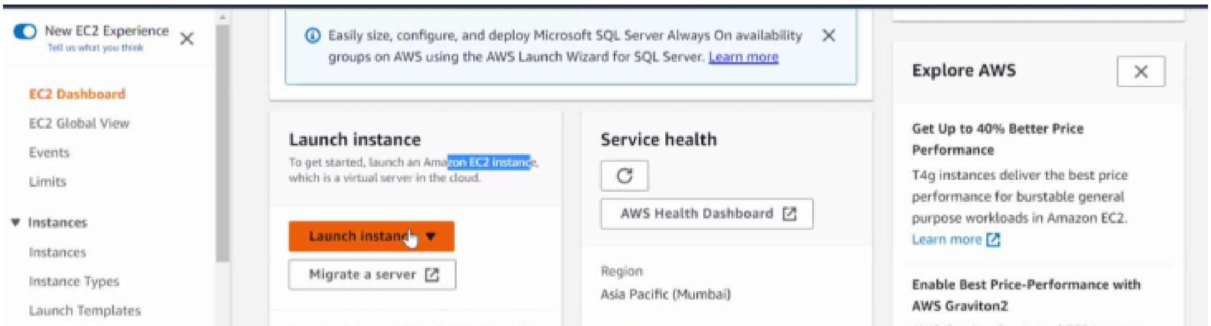
PearlThoughts

Learn AWS core concepts and Terraform **Date: 03/02/2026**



Search results for 'ec2'

Try searching with longer queries for more relevant results



Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

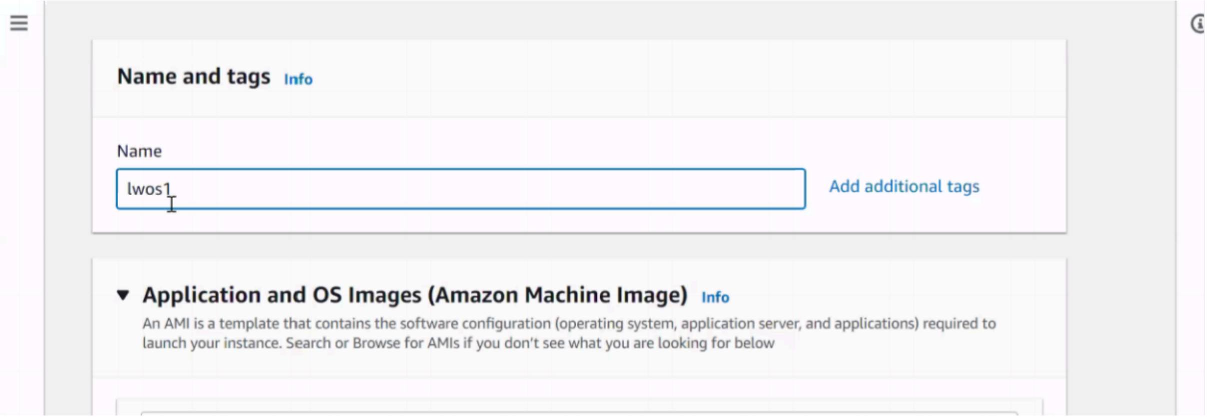
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Region: Asia Pacific (Mumbai)

Get Up to 40% Better Price Performance

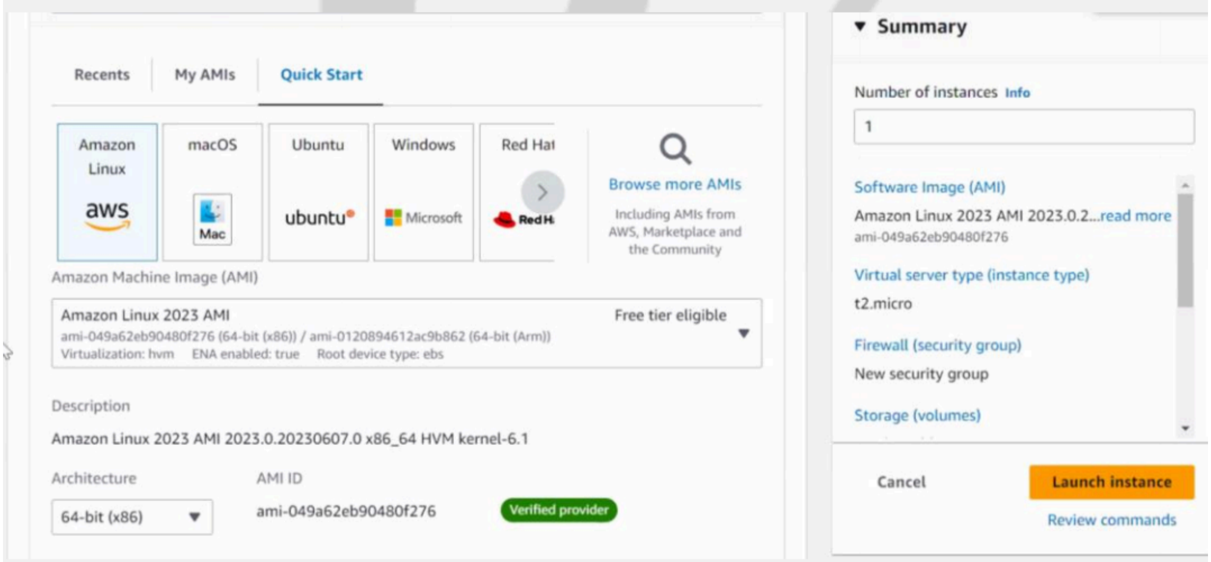
T4g instances deliver the best price performance for burstable general purpose workloads in Amazon EC2. Learn more

Enable Best Price-Performance with AWS Graviton2



Name: lwos1

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below



Amazon Machine Image (AMI)

Amazon Linux 2023 AMI Free tier eligible
ami-049a62eb90480f276 (64-bit (x86)) / ami-0120894612ac9b862 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description: Amazon Linux 2023 AMI 2023.0.20230607.0 x86_64 HVM kernel-6.1

Architecture: 64-bit (x86)

AMI ID: ami-049a62eb90480f276 Verified provider

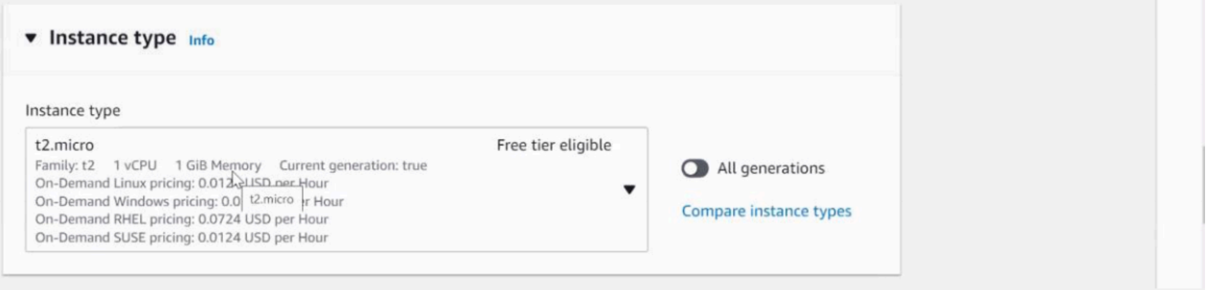
Number of instances Info: 1

Software Image (AMI): Amazon Linux 2023 AMI 2023.0.2...read more
ami-049a62eb90480f276

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

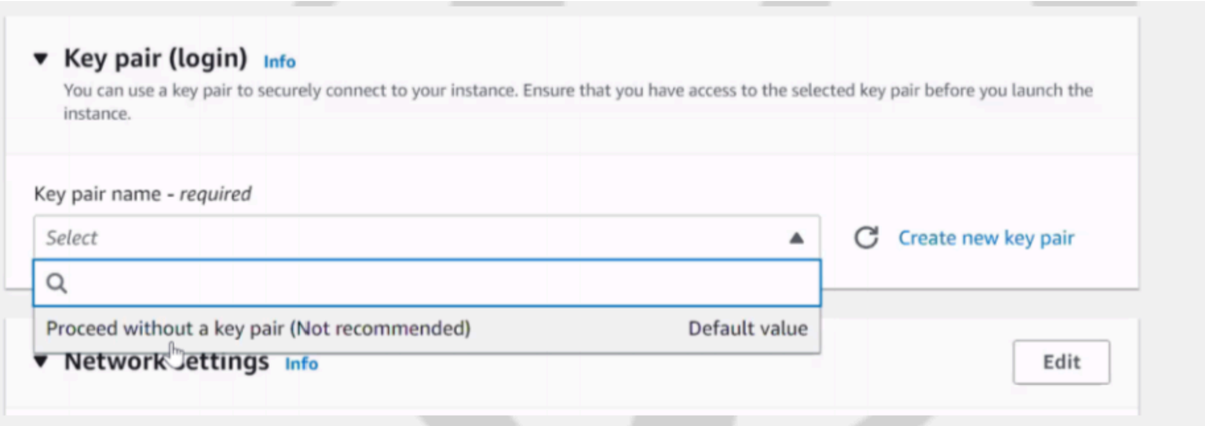
Storage (volumes)



▼ Instance type Info

Instance type

t2.micro Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Linux pricing: 0.012 USD per Hour
On-Demand Windows pricing: 0.0 t2.micro r Hour
On-Demand RHEL pricing: 0.0724 USD per Hour
On-Demand SUSE pricing: 0.0124 USD per Hour



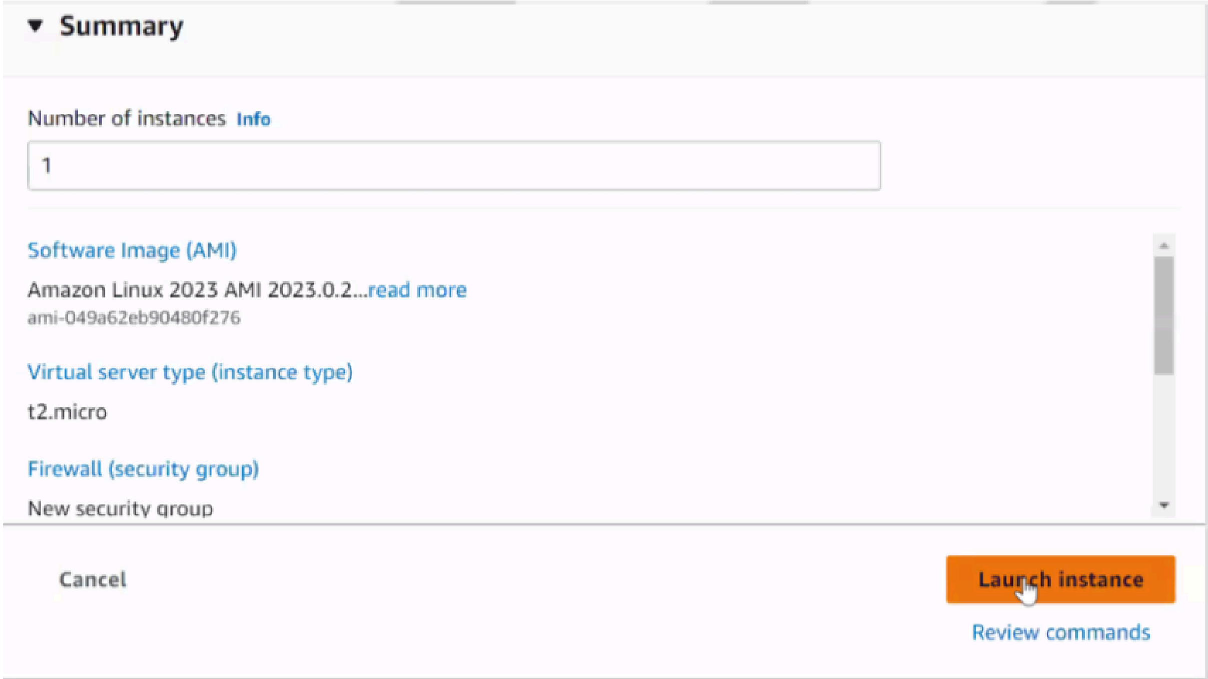
▼ Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

Select

Proceed without a key pair (Not recommended) Default value



▼ Summary

Number of instances Info

1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.0.2...read more
ami-049a62eb90480f276

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Verification

- Instance state: **Running**

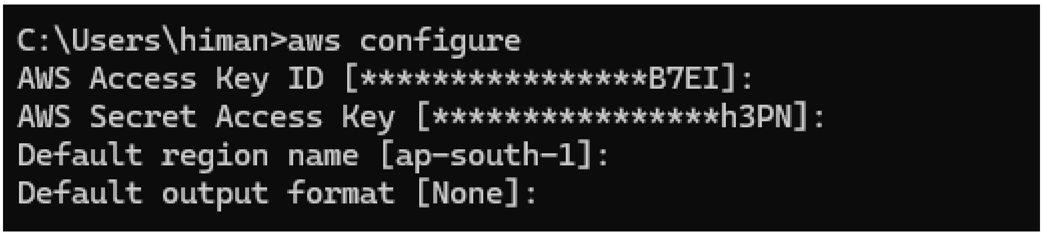
Part 2: Launch EC2 Instance Using Terraform

Prerequisites

- AWS Account
- IAM User with programmatic access
- AWS CLI configured
- Terraform installed

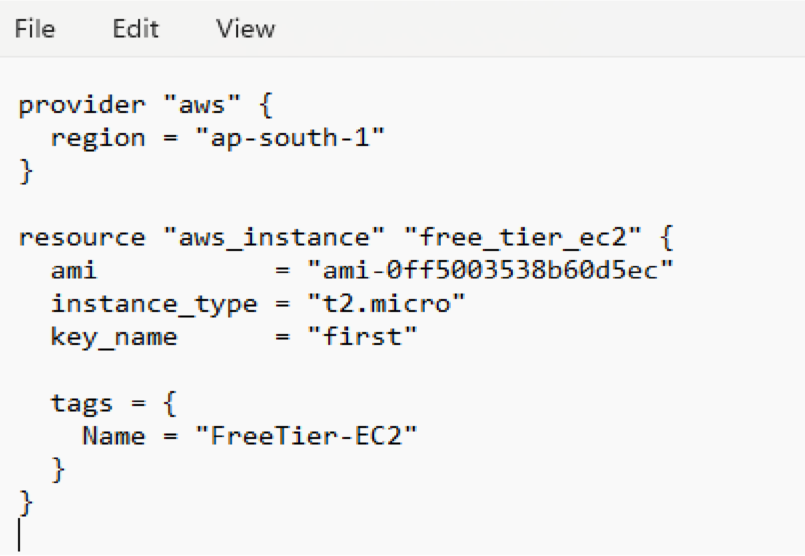
Configure AWS CLI

aws configure



```
C:\Users\himan>aws configure
AWS Access Key ID [****************B7EI]:
AWS Secret Access Key [****************h3PN]:
Default region name [ap-south-1]:
Default output format [None]:
```

Terraform Code for Launching EC2 Instance



```
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "free_tier_ec2" {
  ami           = "ami-0ff5003538b60d5ec"
  instance_type = "t2.micro"
  key_name      = "first"

  tags = {
    Name = "FreeTier-EC2"
  }
}
```

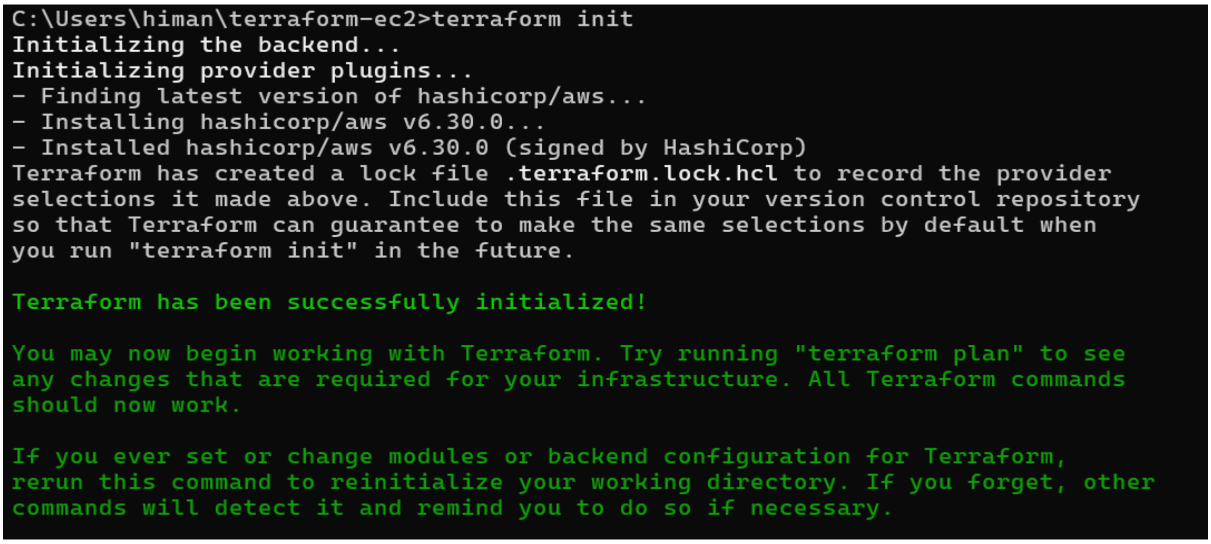
Task:2 **Himansh Sharma**

PearlThoughts

Learn AWS core concepts and Terraform **Date: 03/02/2026**

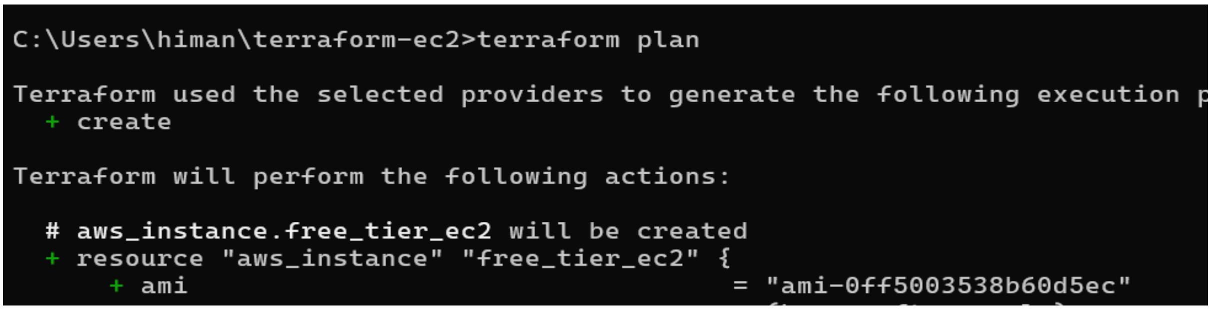
---

1. terraform init



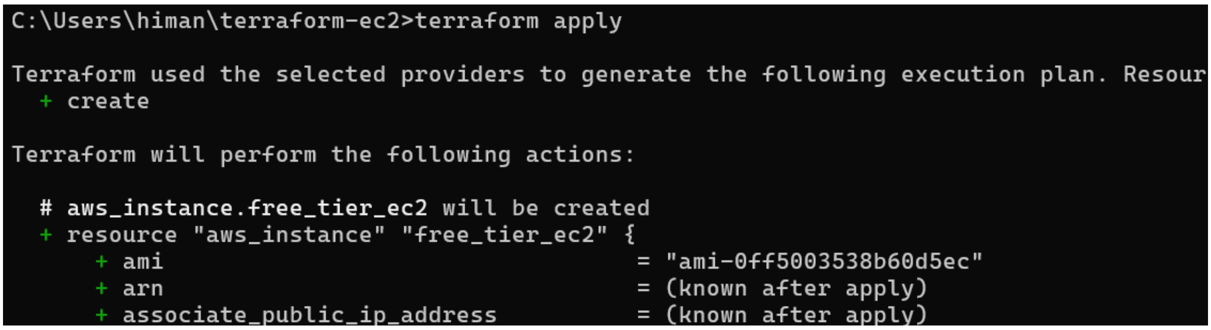
```
C:\Users\himan\terraform-ec2>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.30.0...
- Installed hashicorp/aws v6.30.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl 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.
```

2. terraform plan

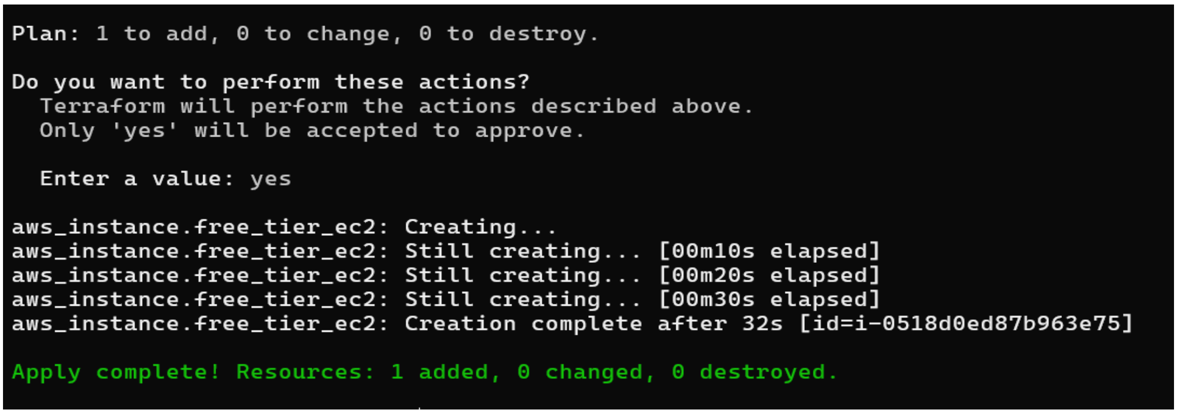


```
C:\Users\himan\terraform-ec2>terraform plan

Terraform used the selected providers to generate the following execution p
  + create

Terraform will perform the following actions:

  # aws_instance.free_tier_ec2 will be created
  + resource "aws_instance" "free_tier_ec2" {
      + ami                                    = "ami-0ff5003538b60d5ec"
```

3. terraform apply



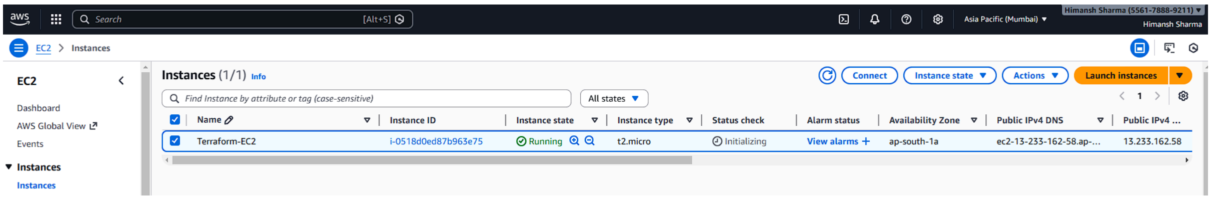
```
C:\Users\himan\terraform-ec2>terraform apply

Terraform used the selected providers to generate the following execution plan. Resour
  + create

Terraform will perform the following actions:

  # aws_instance.free_tier_ec2 will be created
  + resource "aws_instance" "free_tier_ec2" {
      + ami                                    = "ami-0ff5003538b60d5ec"
      + arn                                    = (known after apply)
      + associate_public_ip_address            = (known after apply)
```



```
Plan: 1 to add, 0 to change, 0 to destroy.

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

aws_instance.free_tier_ec2: Creating...
aws_instance.free_tier_ec2: Still creating... [00m10s elapsed]
aws_instance.free_tier_ec2: Still creating... [00m20s elapsed]
aws_instance.free_tier_ec2: Still creating... [00m30s elapsed]
aws_instance.free_tier_ec2: Creation complete after 32s [id=i-0518d0ed87b963e75]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Task:2 **Himansh Sharma**
PearlThoughts
Learn AWS core concepts and Terraform **Date: 03/02/2026**

Output:



Instances (1/1) Info

Find Instance by attribute or tag (case-sensitive)

All states

☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
☑	Terraform-EC2	i-0518d0ed87b963e75	Running	t2.micro	Initializing	View alarms +	ap-south-1a	ec2-13-233-162-58.ap-...	13.233.162.58

Conclusion

- Successfully launched EC2 manually using AWS Console
- Provisioned EC2 automatically using Terraform
- Understood AWS core concepts and Infrastructure as Code

GitHub Repository

- Pushed this project to GitHub