Real-Time Walkthrough to Create the Kubernetes Cluster Using Terraform Script

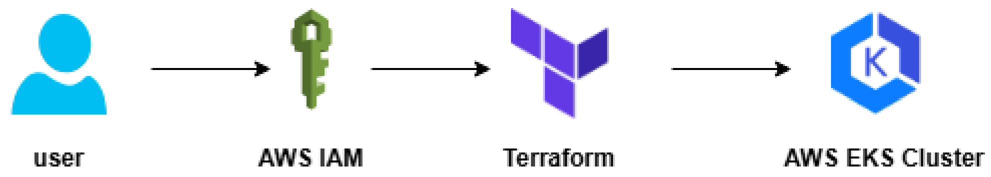
This documentation provides a step-by-step guide to creating an Amazon Elastic Kubernetes Service (EKS) cluster using Terraform, with a focus on addressing the omission of the IAM user. The steps include an explanation of the Terraform script and necessary modifications

Prerequisites

Required Tools:

- Terraform installed on your local system.
- AWS CLI configured with your credentials.

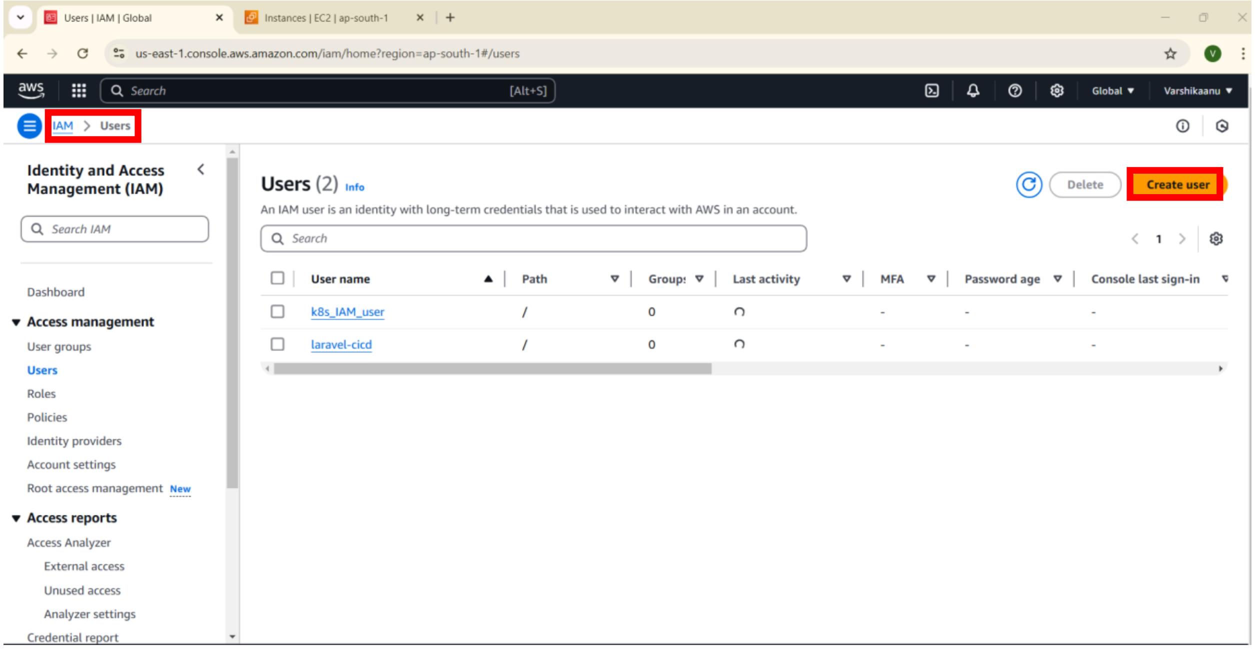
Architecture:



1. Folder Structure

```
project-root/
├── main.tf          # Core Terraform configuration
├── variables.tf     # Input variable definitions
├── outputs.tf       # Output definitions
└── README.md        # Documentation
```

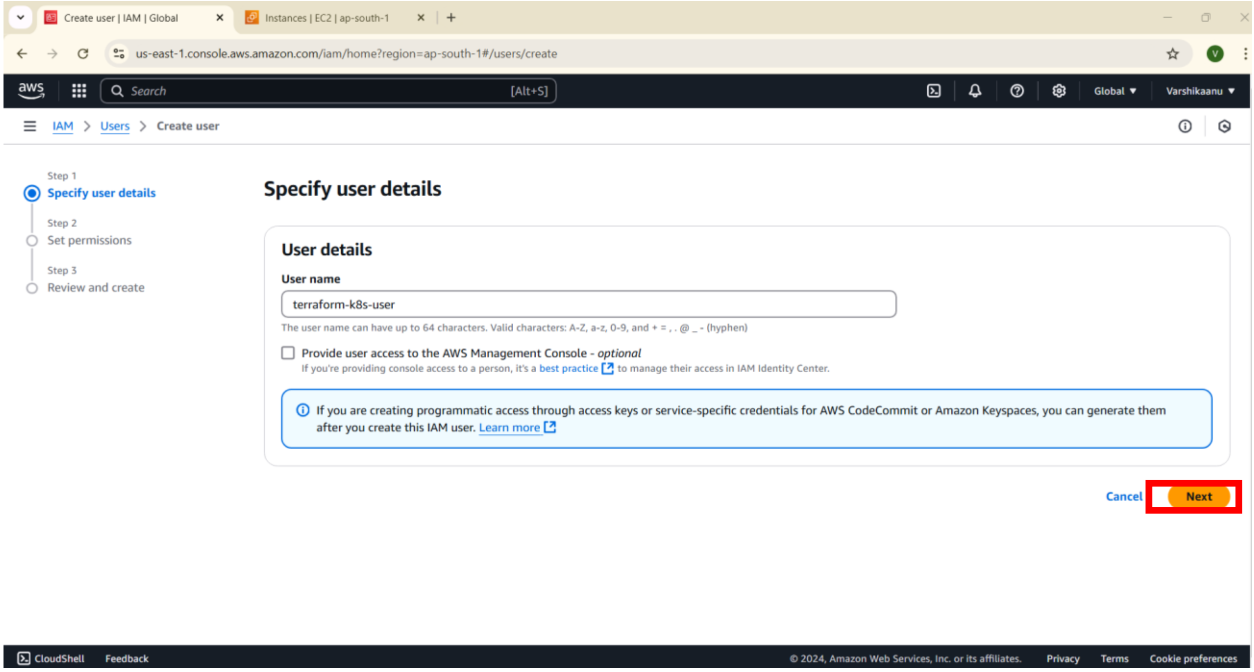
2. IAM User Creation:



Users (2) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

□	User name	Path	Group:	Last activity	MFA	Password age	Console last sign-in
□	k8s_IAM_user	/	0	∩	-	-	-
□	laravel-cicd	/	0	∩	-	-	-


Create user | IAM | Global
Instances | EC2 | ap-south-1
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
Search [Alt+S]
Global ▼
Varshikaanu ▼
IAM > Users > Create user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create
Cancel
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Specify user details

User details

User name

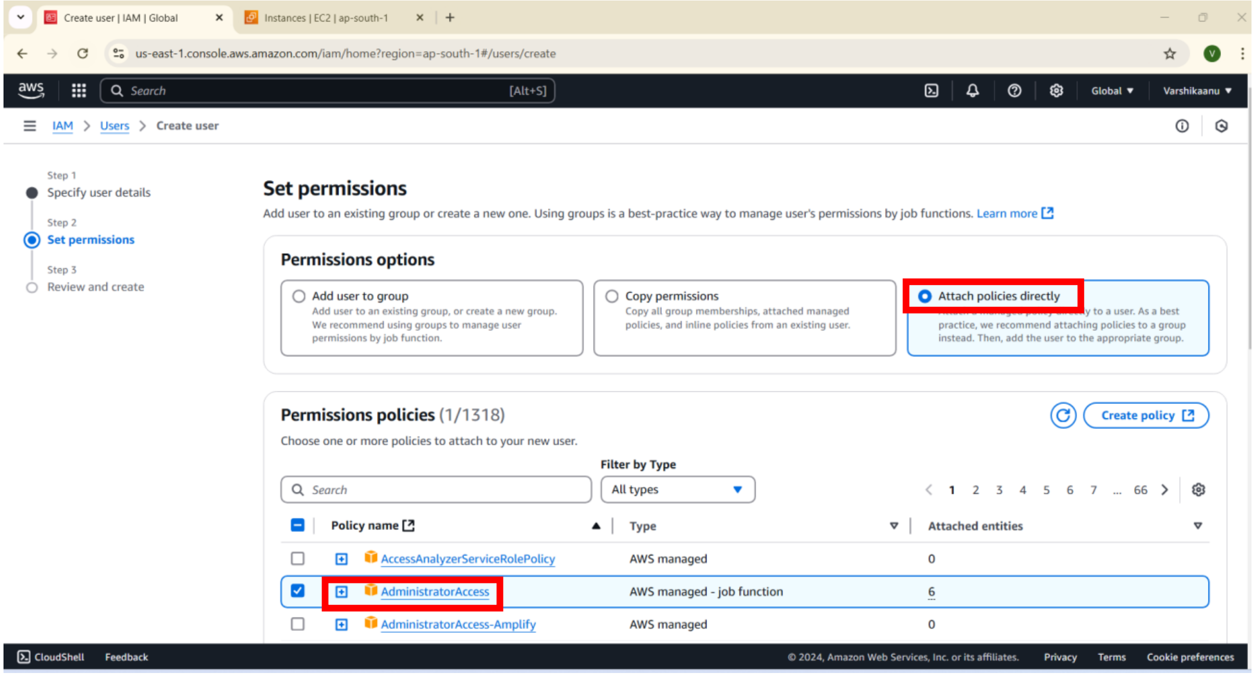
terraform-k8s-user

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

□ Provide user access to the AWS Management Console - *optional*

If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more



Create user | IAM | Global
Instances | EC2 | ap-south-1
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
aws
Search
[Alt+S]
Global ▾
Varshikaanu ▾
IAM > Users > Create user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create

Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more

Permissions options

○ Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

○ Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.

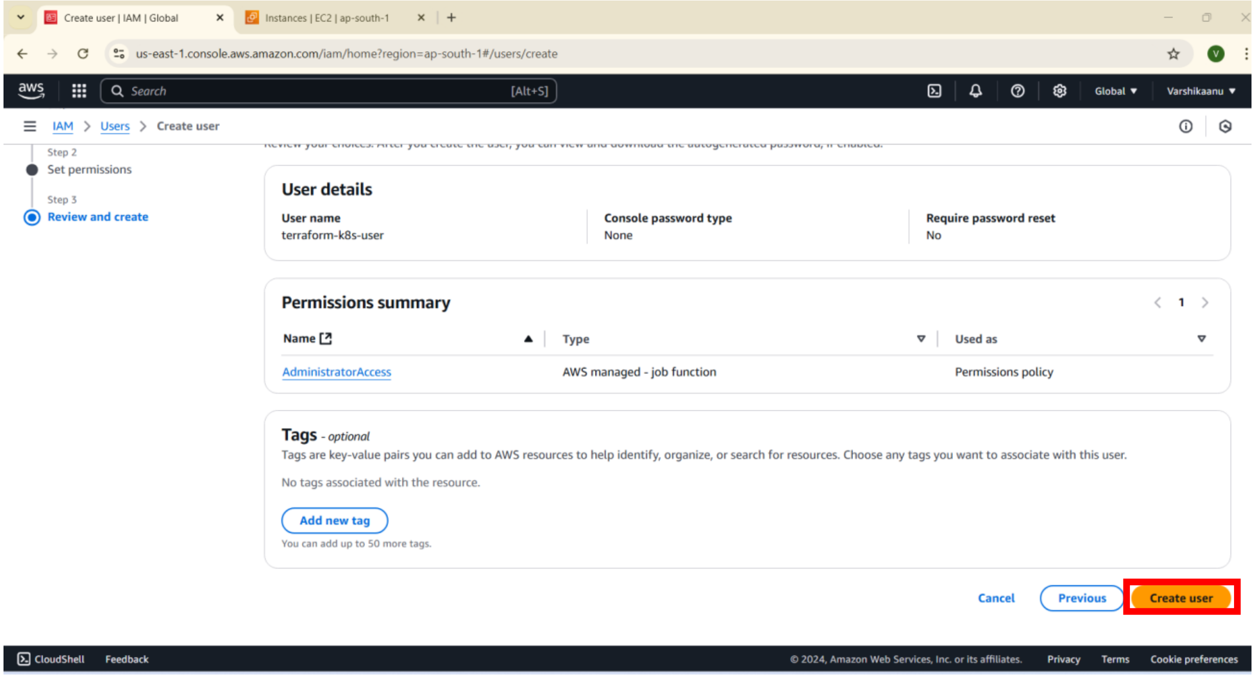
● Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (1/1318)

Choose one or more policies to attach to your new user.

	Policy name	Type	Attached entities
□	AccessAnalyzerServiceRolePolicy	AWS managed	0
☑	AdministratorAccess	AWS managed - job function	6
□	AdministratorAccess-Amplify	AWS managed	0

CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences



Create user | IAM | Global
Instances | EC2 | ap-south-1
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
aws
Search
[Alt+S]
Global ▾
Varshikaanu ▾
IAM > Users > Create user
Step 2
Set permissions
Step 3
Review and create

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details

User name	Console password type	Require password reset
terraform-k8s-user	None	No

Permissions summary

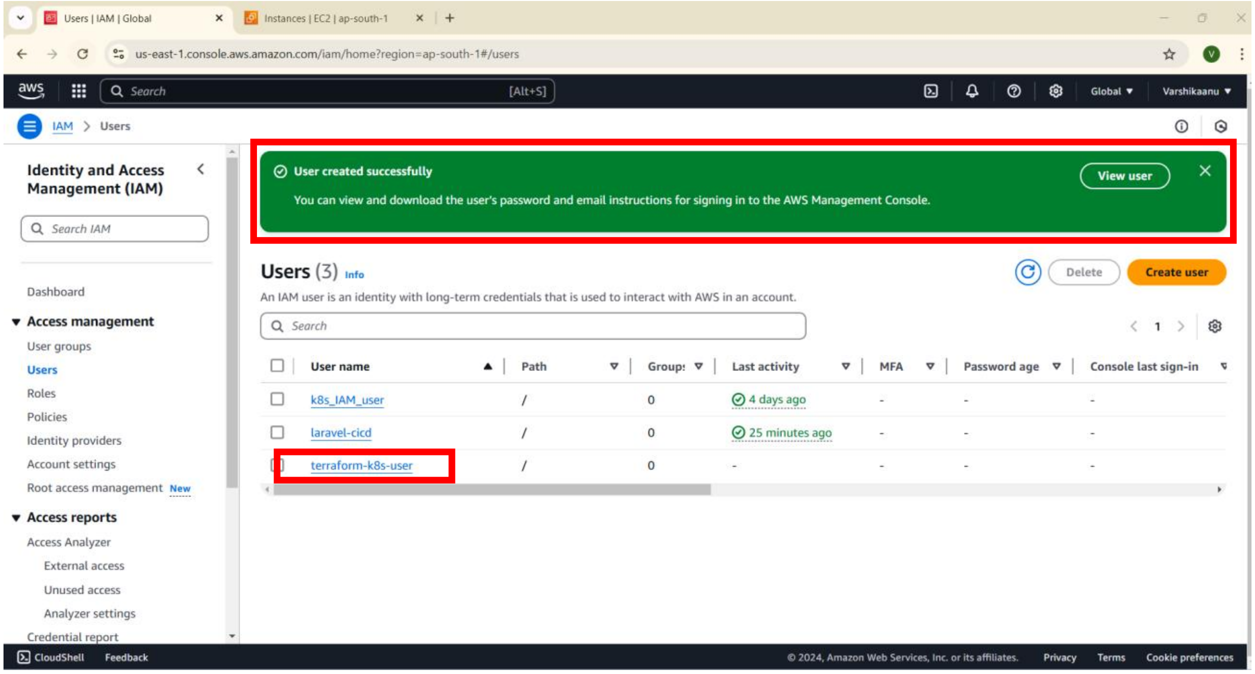
Name	Type	Used as
AdministratorAccess	AWS managed - job function	Permissions policy

Tags - *optional*

Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

You can add up to 50 more tags.



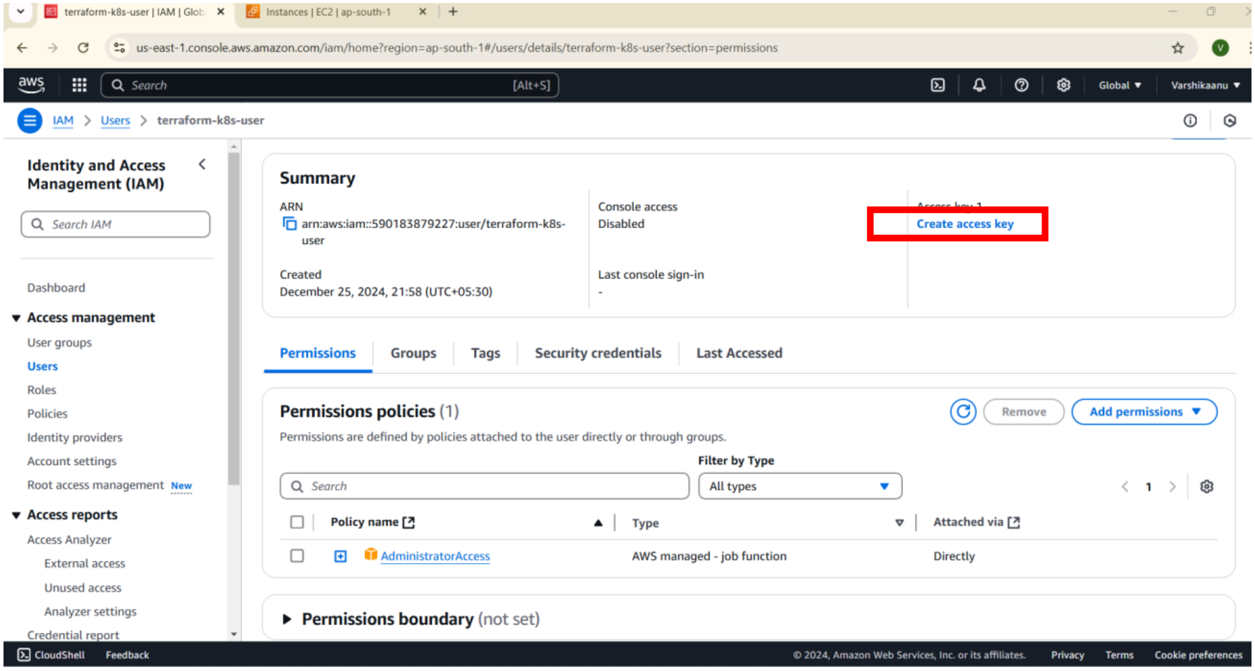
User created successfully

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

Users (3) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

□	User name	Path	Groups	Last activity	MFA	Password age	Console last sign-in
□	k8s_IAM_user	/	0	✓ 4 days ago	-	-	-
□	laravel-cicd	/	0	✓ 25 minutes ago	-	-	-
□	terraform-k8s-user	/	0	-	-	-	-



terraform-k8s-user | IAM | Glob… | Instances | EC2 | ap-south-1

us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/terraform-k8s-user?section=permissions

IAM > Users > terraform-k8s-user

Summary

ARN arn:aws:iam::590183879227:user/terraform-k8s-user	Console access Disabled	Access key 1 Create access key
Created December 25, 2024, 21:58 (UTC+05:30)	Last console sign-in -	

Permissions | Groups | Tags | Security credentials | Last Accessed

Permissions policies (1)

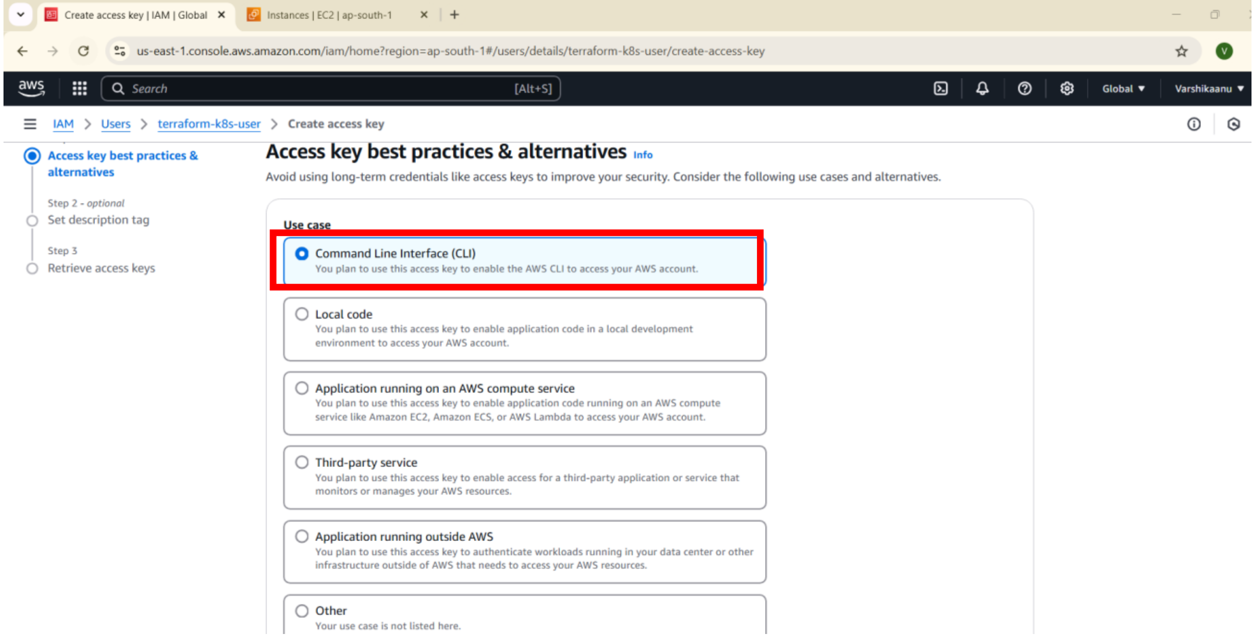
Remove | Add permissions ▼

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type: All types

□	Policy name	Type	Attached via
□	AdministratorAccess	AWS managed - job function	Directly

▶ **Permissions boundary** (not set)



Create access key | IAM | Global | Instances | EC2 | ap-south-1

us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/terraform-k8s-user/create-access-key

IAM > Users > terraform-k8s-user > Create access key

Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

● Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

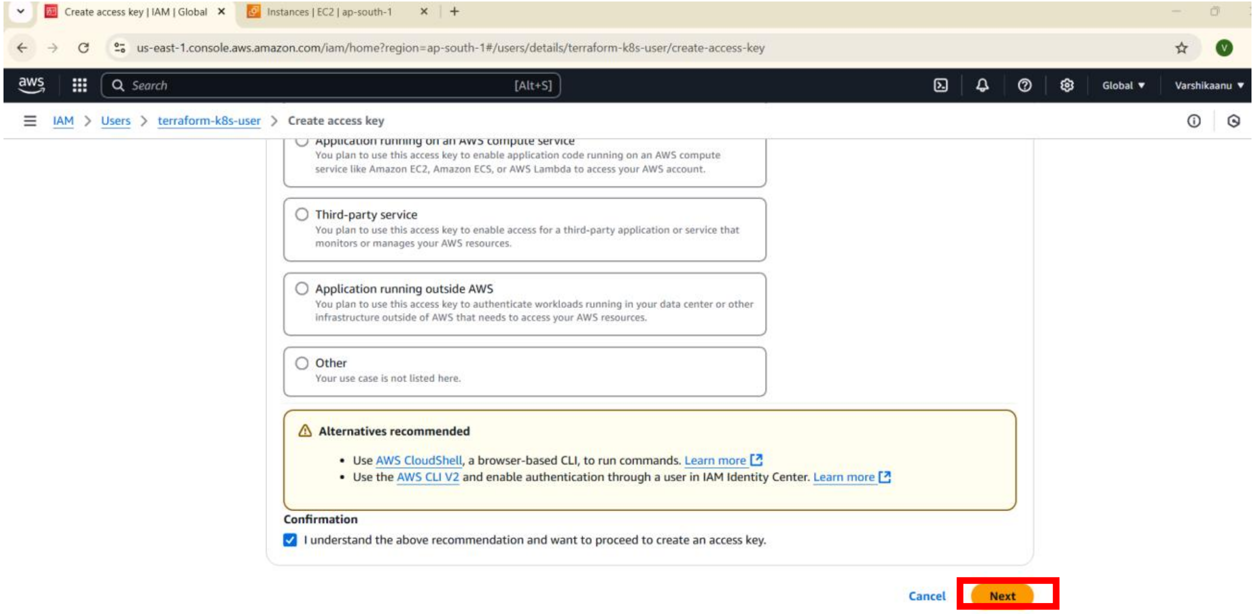
○ Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

○ Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

○ Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

○ Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

○ Other
Your use case is not listed here.


Create access key | IAM | Global
Instances | EC2 | ap-south-1
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/terraform-k8s-user/create-access-key
aws
Search
[Alt+S]
Global ▾
Varshikaanu ▾
IAM > Users > terraform-k8s-user > Create access key

○ Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

○ Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

○ Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

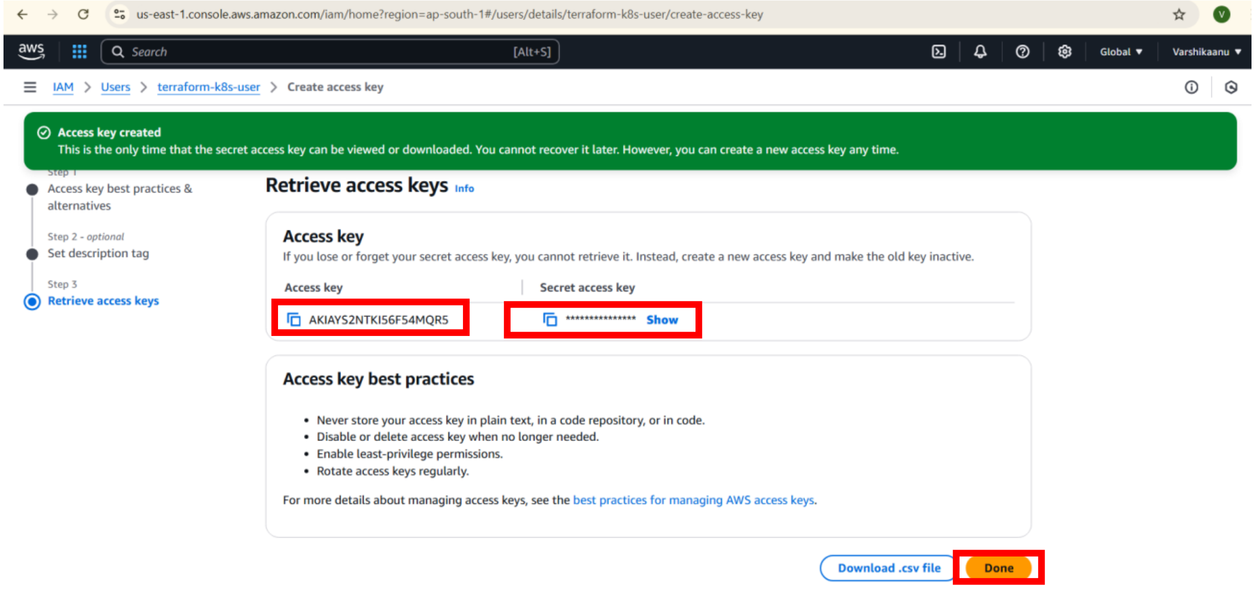
○ Other
Your use case is not listed here.

Alternatives recommended

- Use AWS CloudShell, a browser-based CLI, to run commands. Learn more
- Use the AWS CLI V2 and enable authentication through a user in IAM Identity Center. Learn more

Confirmation

☑ I understand the above recommendation and want to proceed to create an access key.


us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/terraform-k8s-user/create-access-key
aws
Search
[Alt+S]
Global ▾
Varshikaanu ▾
IAM > Users > terraform-k8s-user > Create access key

Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Step 1
Access key best practices & alternatives

Step 2 - *optional*
Set description tag

Step 3
Retrieve access keys

Retrieve access keys Info

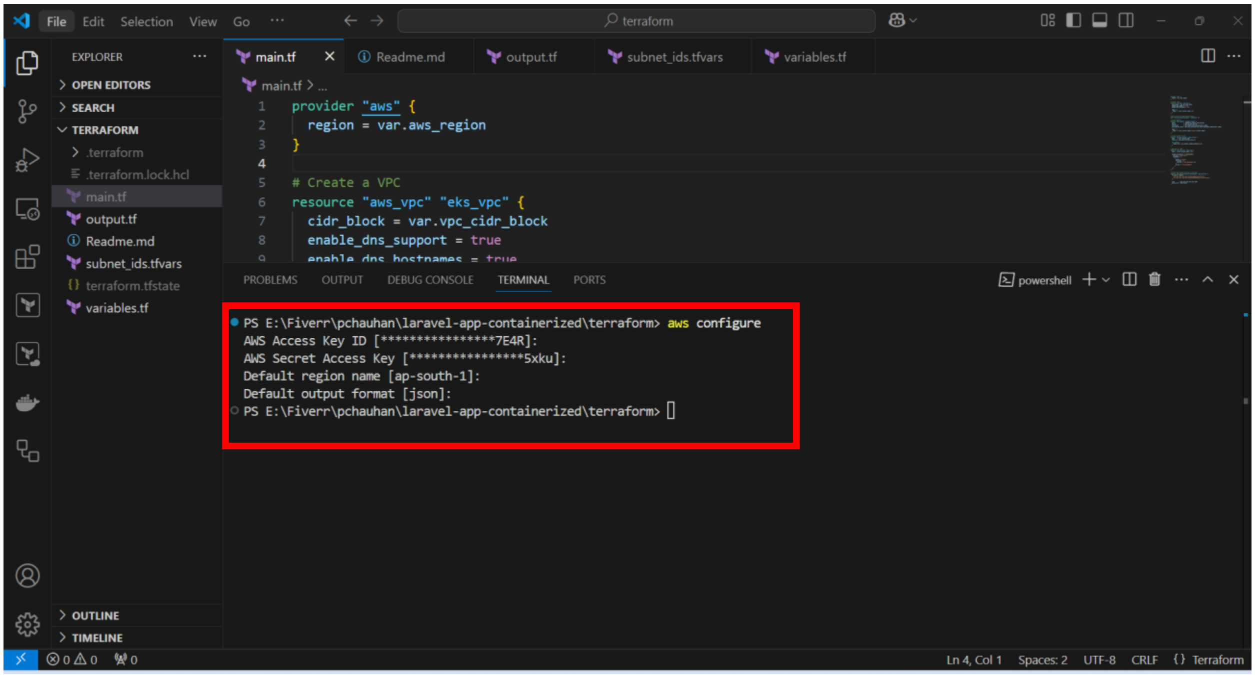
Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIAYS2NTKI56F54MQR5	*************** Show

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the best practices for managing AWS access keys.



```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true
```

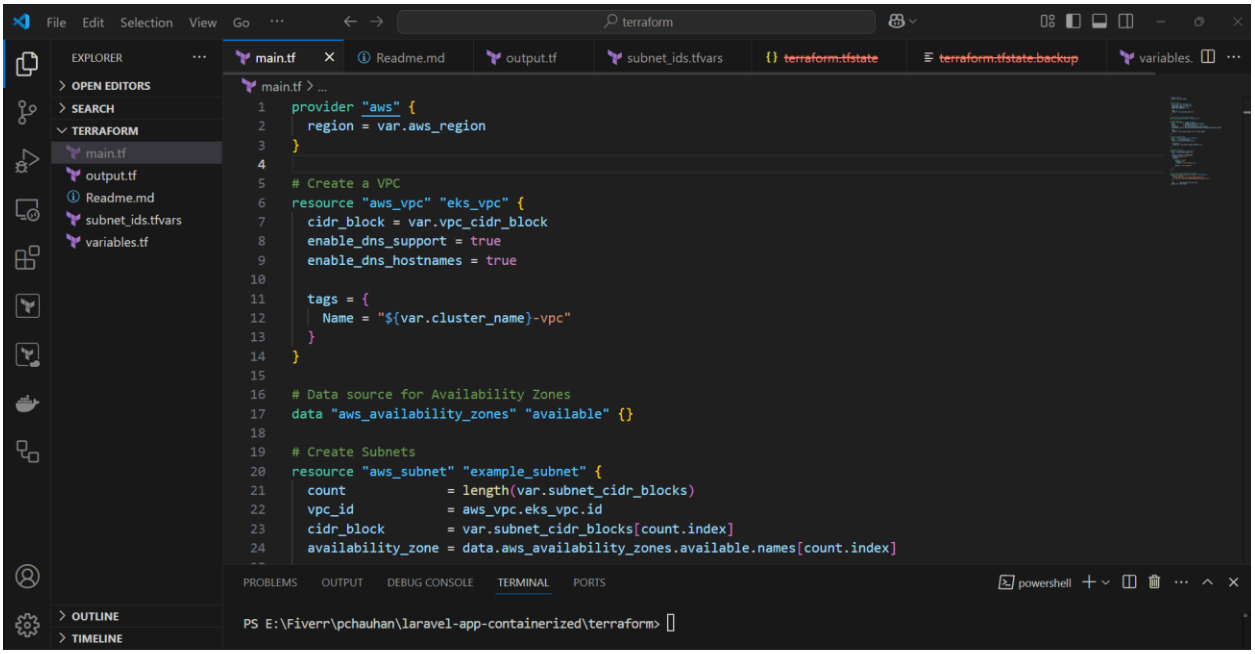
```
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform> aws configure
AWS Access Key ID [****************7E4R]:
AWS Secret Access Key [****************5xku]:
Default region name [ap-south-1]:
Default output format [json]:
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform>
```

2. Terraform Script Overview

Main.tf

```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true

  tags = {
    Name = "${var.cluster_name}-vpc"
  }
}

# Data source for Availability Zones
data "aws_availability_zones" "available" {}

# Create Subnets
resource "aws_subnet" "example_subnet" {
  count                = length(var.subnet_cidr_blocks)
  vpc_id               = aws_vpc.eks_vpc.id
  cidr_block        = var.subnet_cidr_blocks[count.index]
  availability_zone = data.aws_availability_zones.available.names[count.index]

  tags = {
    Name = "${var.cluster_name}-subnet-${count.index}"
  }
}

# Create an EKS Cluster
resource "aws_eks_cluster" "eks_cluster" {
  name     = var.cluster_name
  role_arn = aws_iam_role.eks_role.arn

  vpc_config {
    subnet_ids = aws_subnet.example_subnet[*].id
  }
}

# IAM Role for EKS
resource "aws_iam_role" "eks_role" {
  name = "${var.cluster_name}-role"

  assume_role_policy = jsonencode({
    Version = "2012-10-17"
    Statement = [
      {
        Effect = "Allow"
        Principal = {
          Service = "eks.amazonaws.com"
        }
        Action = "sts:AssumeRole"
      }
    ]
  })
}

# Attach EKS Managed Policies to the Role
resource "aws_iam_role_policy_attachment" "eks_policies" {
  for_each = toset([
    "arn:aws:iam::aws:policy/AmazonEKSClusterPolicy",
    "arn:aws:iam::aws:policy/AmazonEKSVPCResourceController",
  ])

  role       = aws_iam_role.eks_role.name
  policy_arn = each.value
```

```
}
```



```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true

  tags = {
    Name = "${var.cluster_name}-vpc"
  }
}

# Data source for Availability Zones
data "aws_availability_zones" "available" {}

# Create Subnets
resource "aws_subnet" "example_subnet" {
  count             = length(var.subnet_cidr_blocks)
  vpc_id            = aws_vpc.eks_vpc.id
  cidr_block        = var.subnet_cidr_blocks[count.index]
  availability_zone = data.aws_availability_zones.available.names[count.index]
```

```
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform>
```

output.tf

```
output "eks_cluster_name" {
  description = "Name of the EKS cluster"
  value       = aws_eks_cluster.eks_cluster.name
}

output "eks_cluster_endpoint" {
  description = "EKS Cluster endpoint"
  value       = aws_eks_cluster.eks_cluster.endpoint
}

output "eks_cluster_arn" {
  description = "EKS Cluster ARN"
  value       = aws_eks_cluster.eks_cluster.arn
}
```

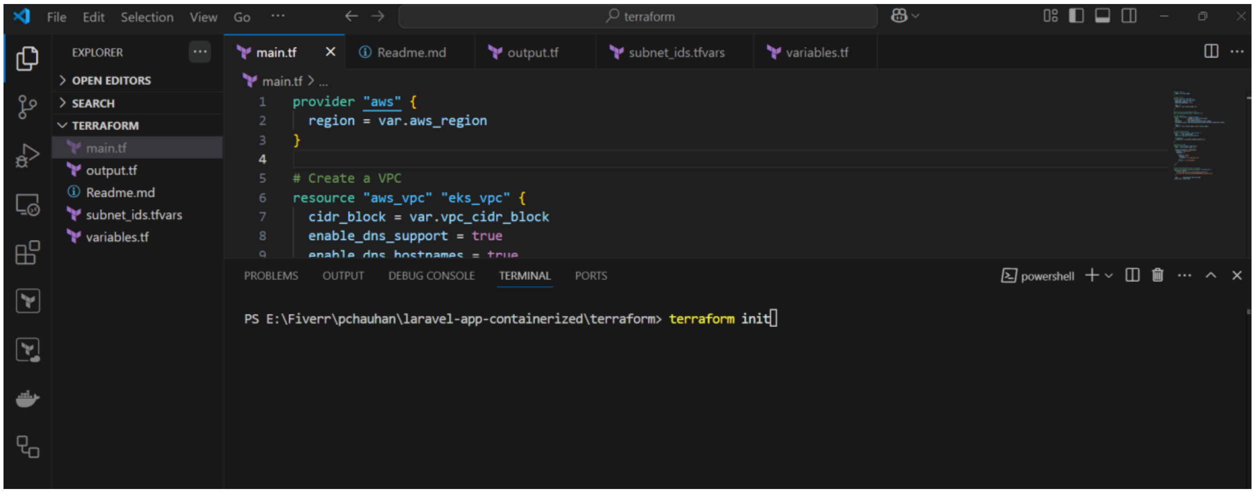
variable.tf

```
variable "aws_region" {
  description = "AWS region to deploy resources"
  default     = "ap-south-1"
}

variable "cluster_name" {
  description = "Name of the EKS cluster"
  default     = "alvin-eks-cluster"
}

# VPC CIDR block for the newly created VPC
variable "vpc_cidr_block" {
  description = "CIDR block for the VPC"
  default     = "10.0.0.0/16"
}

# Subnet CIDR blocks for the VPC
variable "subnet_cidr_blocks" {
  description = "List of CIDR blocks for the subnets"
  type        = list(string)
  default     = ["10.0.1.0/24", "10.0.2.0/24"]
}

# (Optional) VPC ID if you want to specify an existing VPC
variable "vpc_id" {
  description = "ID of the VPC where subnets will be created"
  type        = string
  default     = ""
}
```

3. Execution Steps

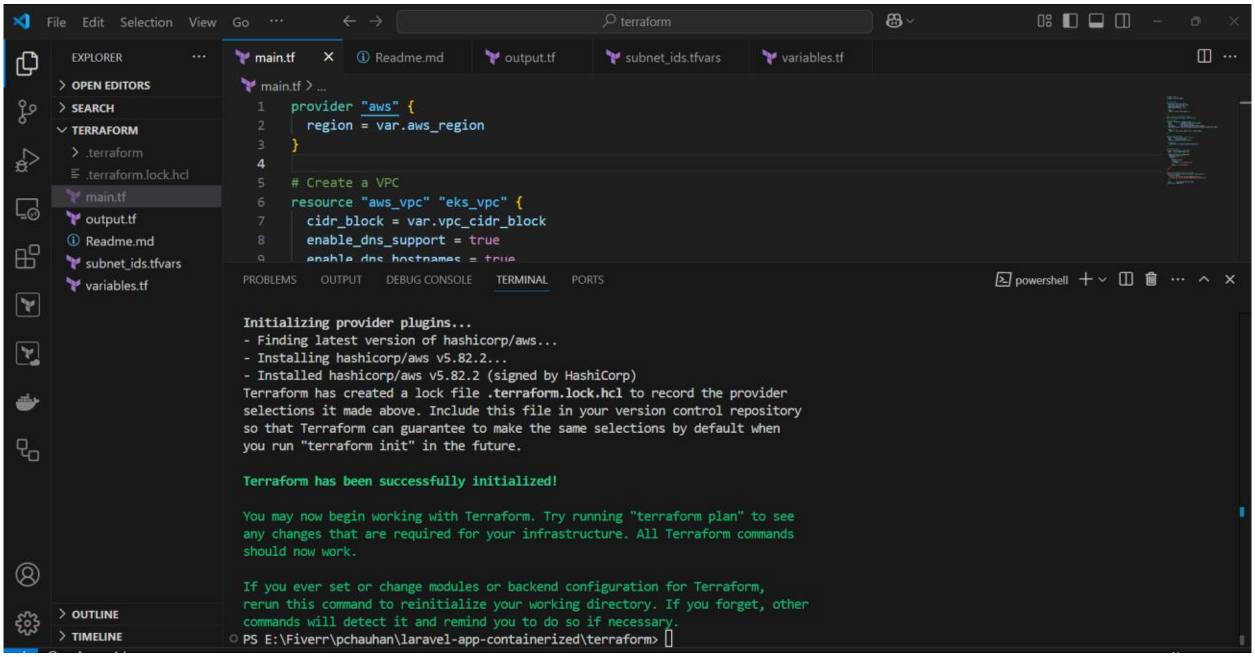
Initialize Terraform:

```
terraform init
```



```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true
```

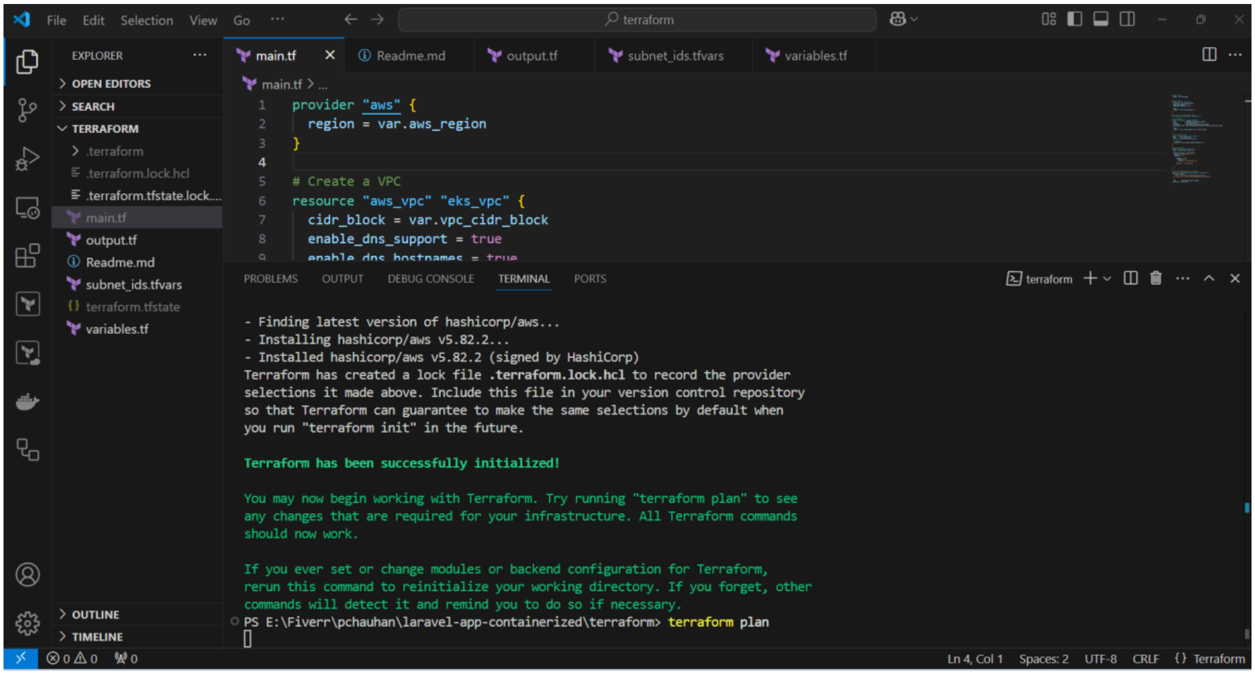
```
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform> terraform init
```



```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true
```

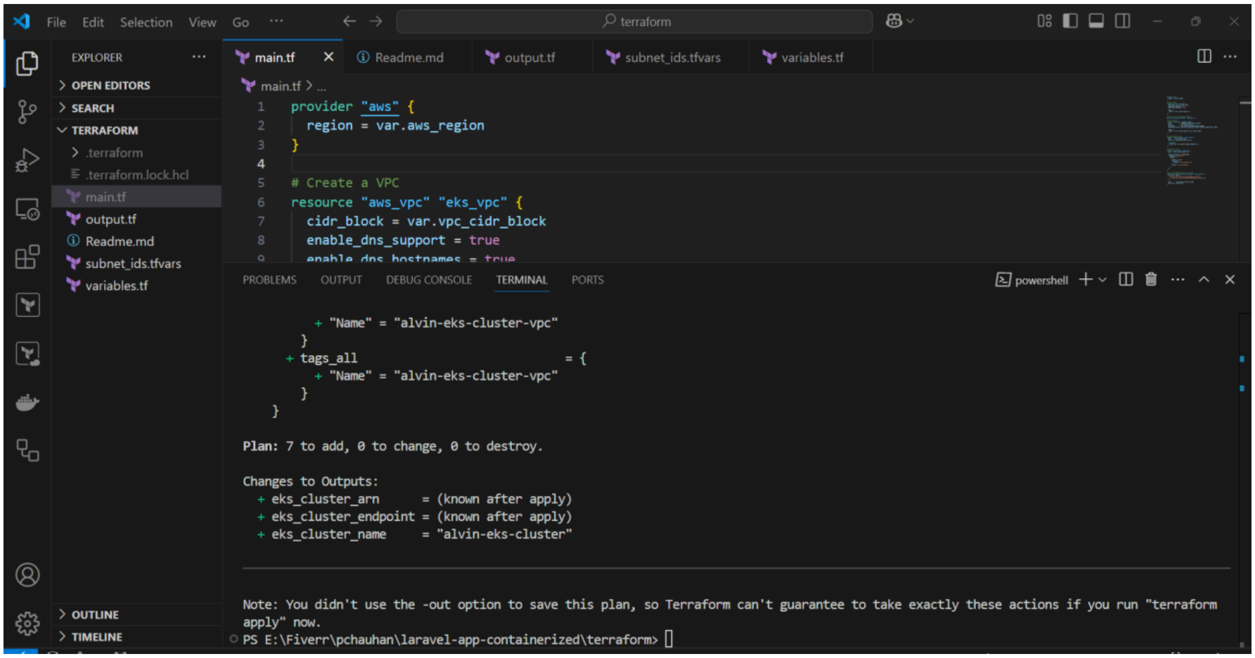
```
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.82.2...
- Installed hashicorp/aws v5.82.2 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your 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.
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform>
```

```
terraform plan
```



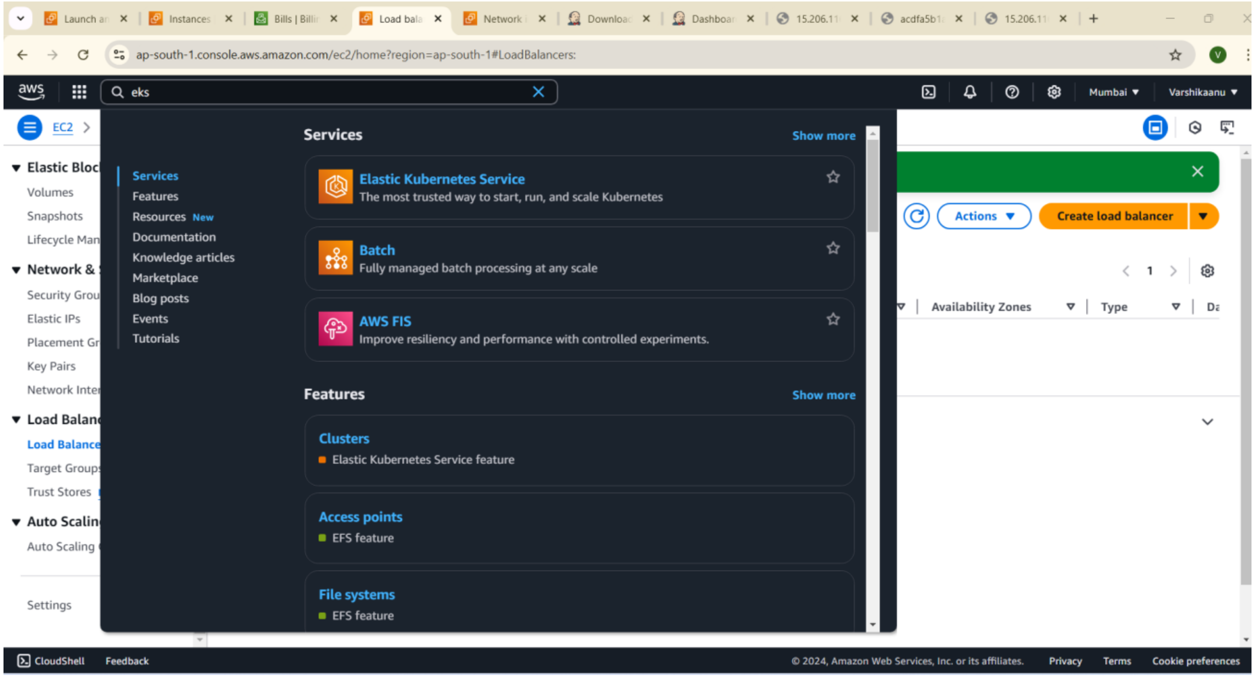
```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true
```

```
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.82.2...
- Installed hashicorp/aws v5.82.2 (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.
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform> terraform plan
```

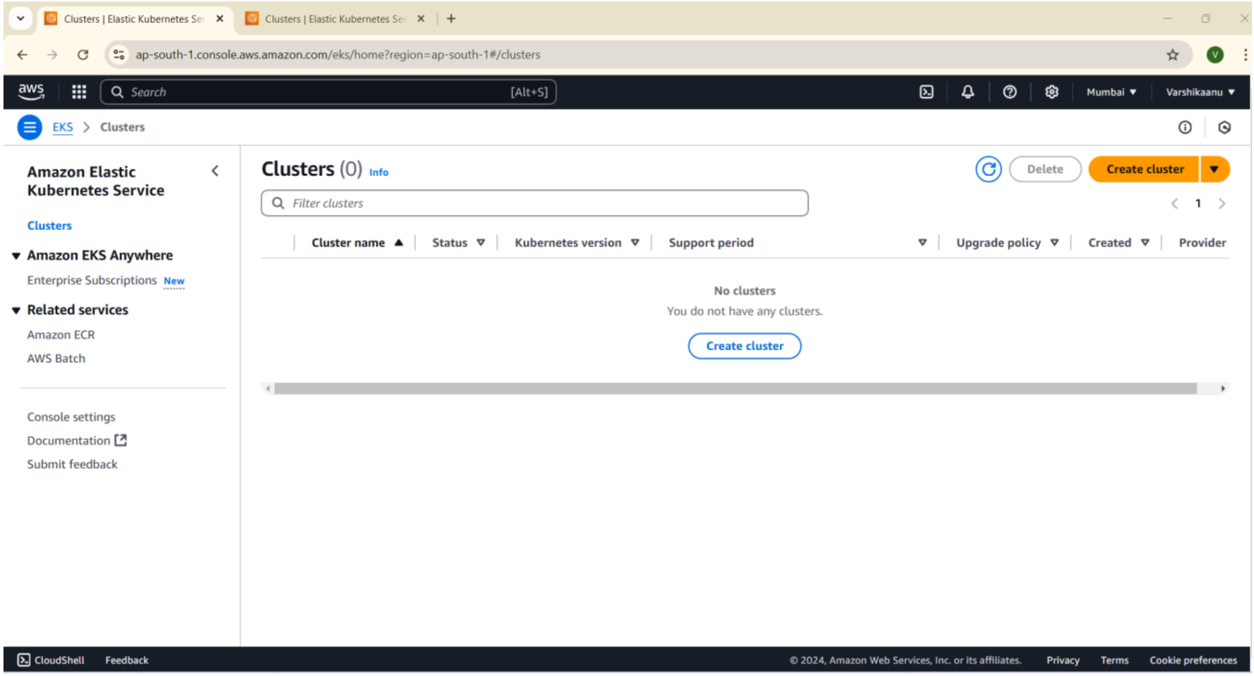


```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true
```

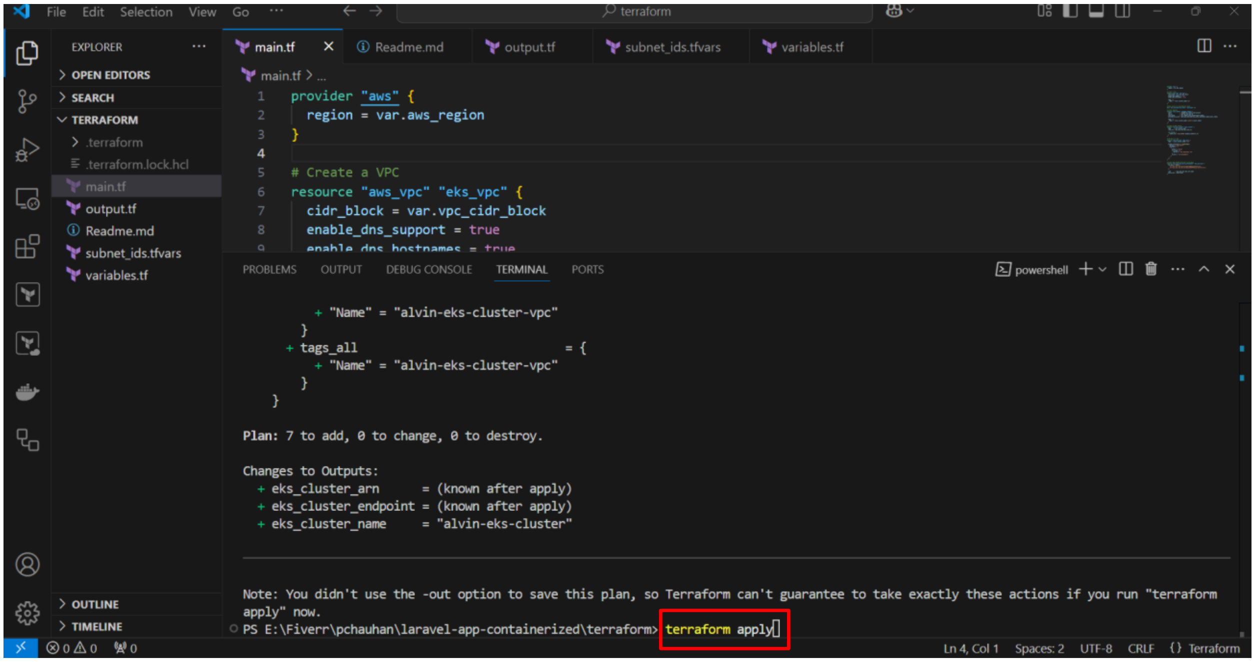
```
          + "Name" = "alvin-eks-cluster-vpc"
        }
      + tags_all                             = {
          + "Name" = "alvin-eks-cluster-vpc"
        }
    }

Plan: 7 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + eks_cluster_arn      = (known after apply)
  + eks_cluster_endpoint = (known after apply)
  + eks_cluster_name     = "alvin-eks-cluster"

─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform
apply" now.
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform>
```


ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#LoadBalancers:
eks
Services
Features
Resources New
Documentation
Knowledge articles
Marketplace
Blog posts
Events
Tutorials
Services
Show more
Elastic Kubernetes Service
The most trusted way to start, run, and scale Kubernetes
Batch
Fully managed batch processing at any scale
AWS FIS
Improve resiliency and performance with controlled experiments.
Features
Show more
Clusters
Elastic Kubernetes Service feature
Access points
EFS feature
File systems
EFS feature
Actions
Create load balancer
Availability Zones
Type
Mumbai
Varshikaanu
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Before executing the terraform apply command, I retrieved and reviewed the EKS output.

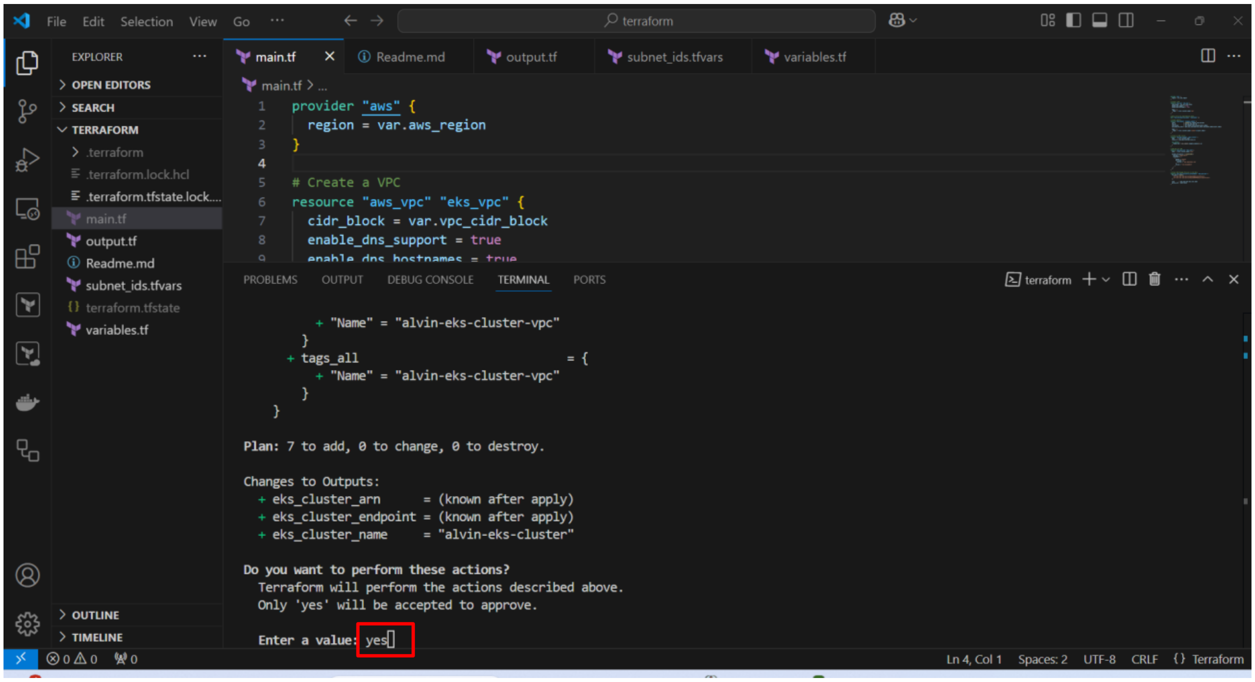


```
terraform apply
```



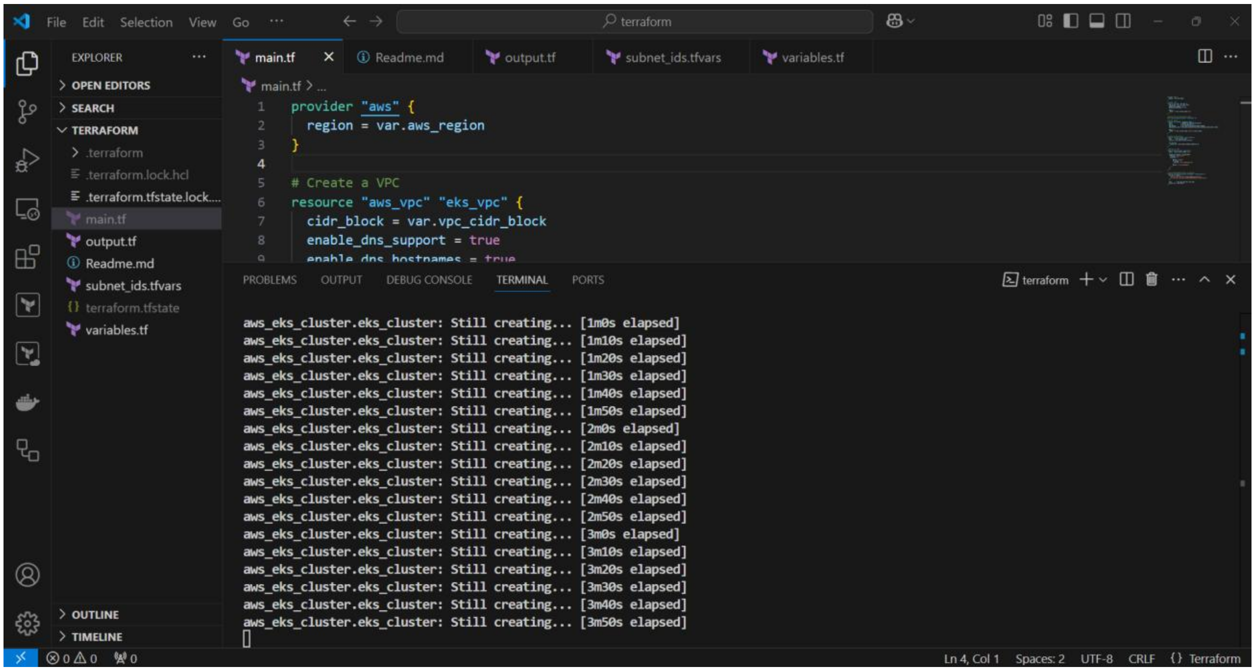
```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true
```

```
          + "Name" = "alvin-eks-cluster-vpc"
        }
      + tags_all                             = {
          + "Name" = "alvin-eks-cluster-vpc"
        }
    }

Plan: 7 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + eks_cluster_arn      = (known after apply)
  + eks_cluster_endpoint = (known after apply)
  + eks_cluster_name     = "alvin-eks-cluster"

─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform
apply" now.
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform> terraform apply
```



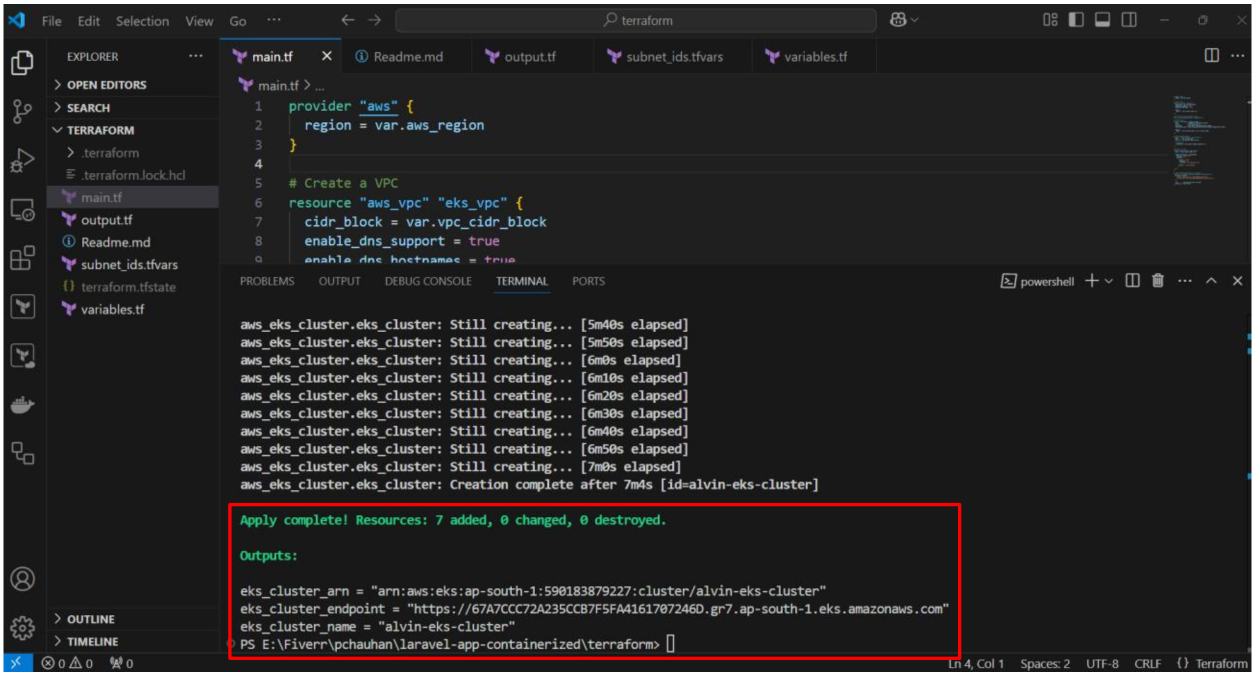
```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true
```

```
          + "Name" = "alvin-eks-cluster-vpc"
        }
      + tags_all                             = {
          + "Name" = "alvin-eks-cluster-vpc"
        }
    }

Plan: 7 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + eks_cluster_arn      = (known after apply)
  + eks_cluster_endpoint = (known after apply)
  + eks_cluster_name     = "alvin-eks-cluster"

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
```



```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true
```

```
aws_eks_cluster.eks_cluster: Still creating... [1m0s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [1m10s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [1m20s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [1m30s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [1m40s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [1m50s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [2m0s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [2m10s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [2m20s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [2m30s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [2m40s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [2m50s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [3m0s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [3m10s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [3m20s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [3m30s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [3m40s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [3m50s elapsed]
```



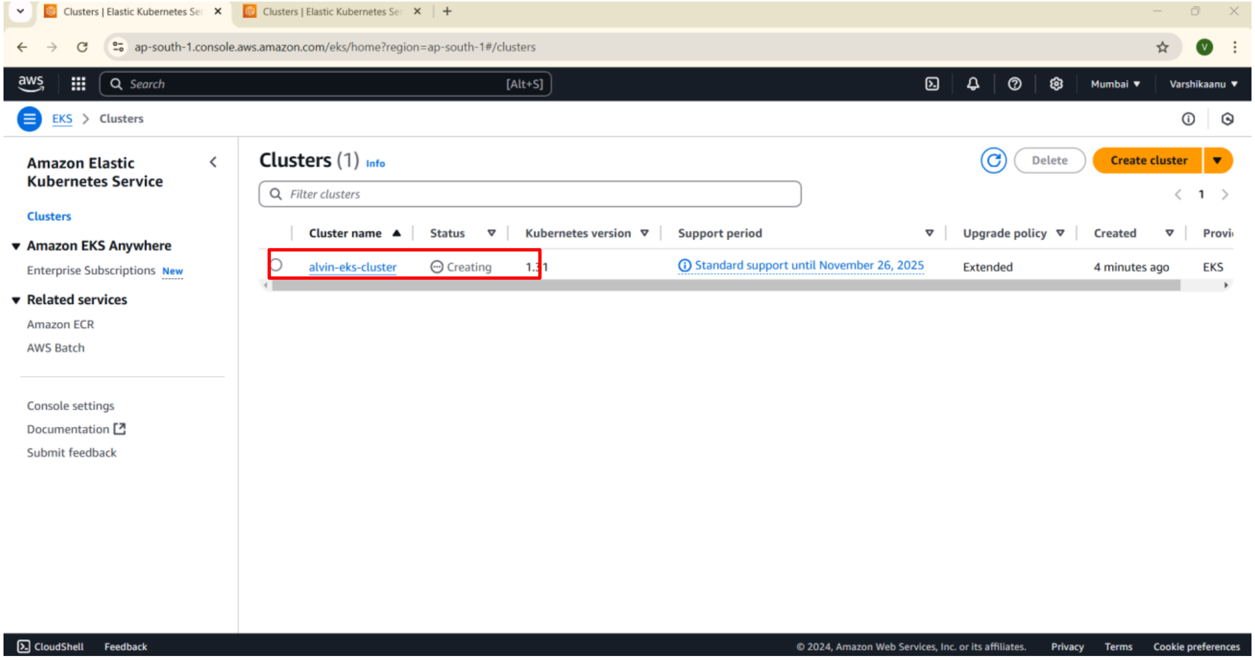
```
provider "aws" {
  region = var.aws_region
}

# Create a VPC
resource "aws_vpc" "eks_vpc" {
  cidr_block = var.vpc_cidr_block
  enable_dns_support = true
  enable_dns_hostnames = true
```

```
aws_eks_cluster.eks_cluster: Still creating... [5m40s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [5m50s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [6m0s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [6m10s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [6m20s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [6m30s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [6m40s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [6m50s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [7m0s elapsed]
aws_eks_cluster.eks_cluster: Creation complete after 7m4s [id=alvin-eks-cluster]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

Outputs:

eks_cluster_arn = "arn:aws:eks:ap-south-1:590183879227:cluster/alvin-eks-cluster"
eks_cluster_endpoint = "https://67A7CCC72A235CCB7F5FA4161707246D.gr7.ap-south-1.eks.amazonaws.com"
eks_cluster_name = "alvin-eks-cluster"
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform>
```

To validate the EKS cluster:

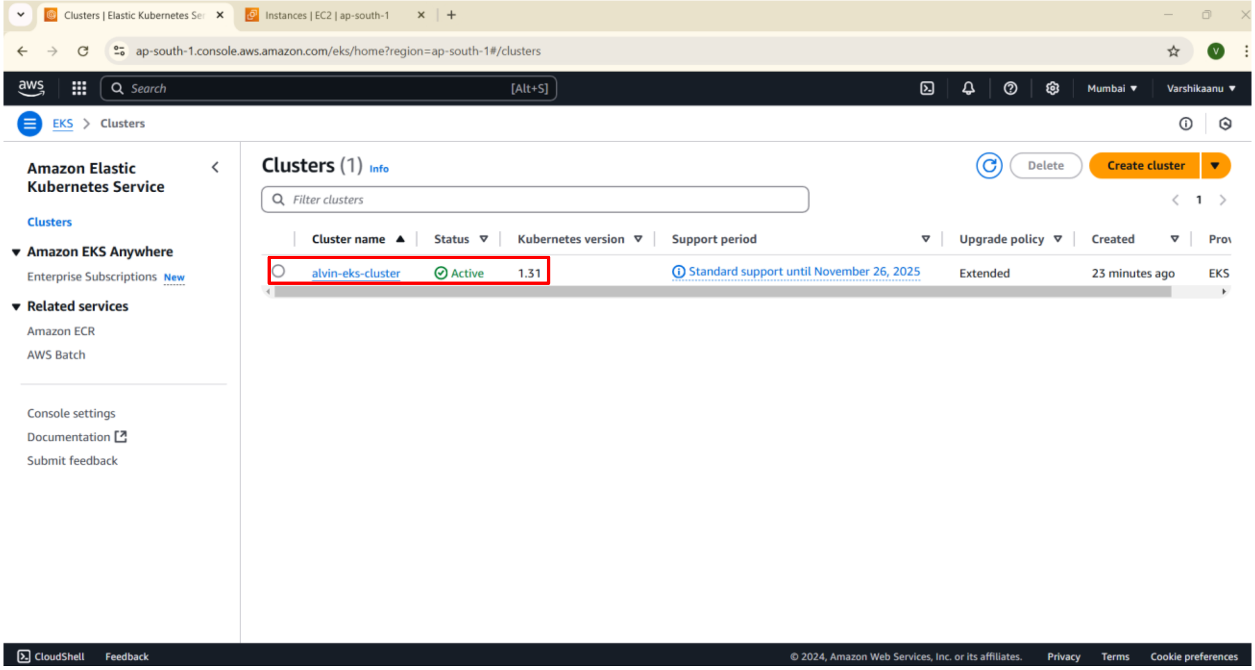
1. Navigate to the **Amazon EKS console**.

2. Review the cluster's status and configuration details to ensure it has been created successfully.


Clusters | Elastic Kubernetes Se
Clusters | Elastic Kubernetes Se
ap-south-1.console.aws.amazon.com/eks/home?region=ap-south-1#/clusters
Search
[Alt+S]
Mumbai
Varshikaanu
EKS > Clusters
Amazon Elastic Kubernetes Service
Clusters
Amazon EKS Anywhere
Enterprise Subscriptions New
Related services
Amazon ECR
AWS Batch
Console settings
Documentation
Submit feedback
Delete
Create cluster
Filter clusters
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Clusters (1) Info

Cluster name	Status	Kubernetes version	Support period	Upgrade policy	Created	Provi
alvin-eks-cluster	Creating	1.31	Standard support until November 26, 2025	Extended	4 minutes ago	EKS


Clusters | Elastic Kubernetes Se
Instances | EC2 | ap-south-1
ap-south-1.console.aws.amazon.com/eks/home?region=ap-south-1#/clusters
Search
[Alt+S]
Mumbai
Varshikaanu
EKS > Clusters
Amazon Elastic Kubernetes Service
Clusters
Amazon EKS Anywhere
Enterprise Subscriptions New
Related services
Amazon ECR
AWS Batch
Console settings
Documentation
Submit feedback
Delete
Create cluster
Filter clusters
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

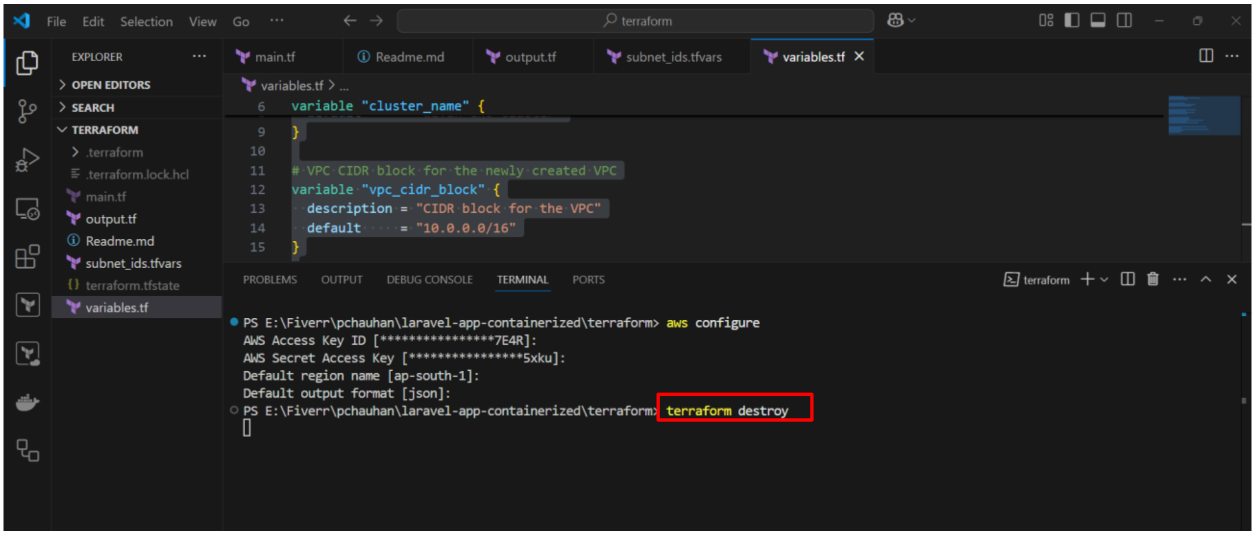
Clusters (1) Info

Cluster name	Status	Kubernetes version	Support period	Upgrade policy	Created	Prov
alvin-eks-cluster	Active	1.31	Standard support until November 26, 2025	Extended	23 minutes ago	EKS

To destroy the resources:

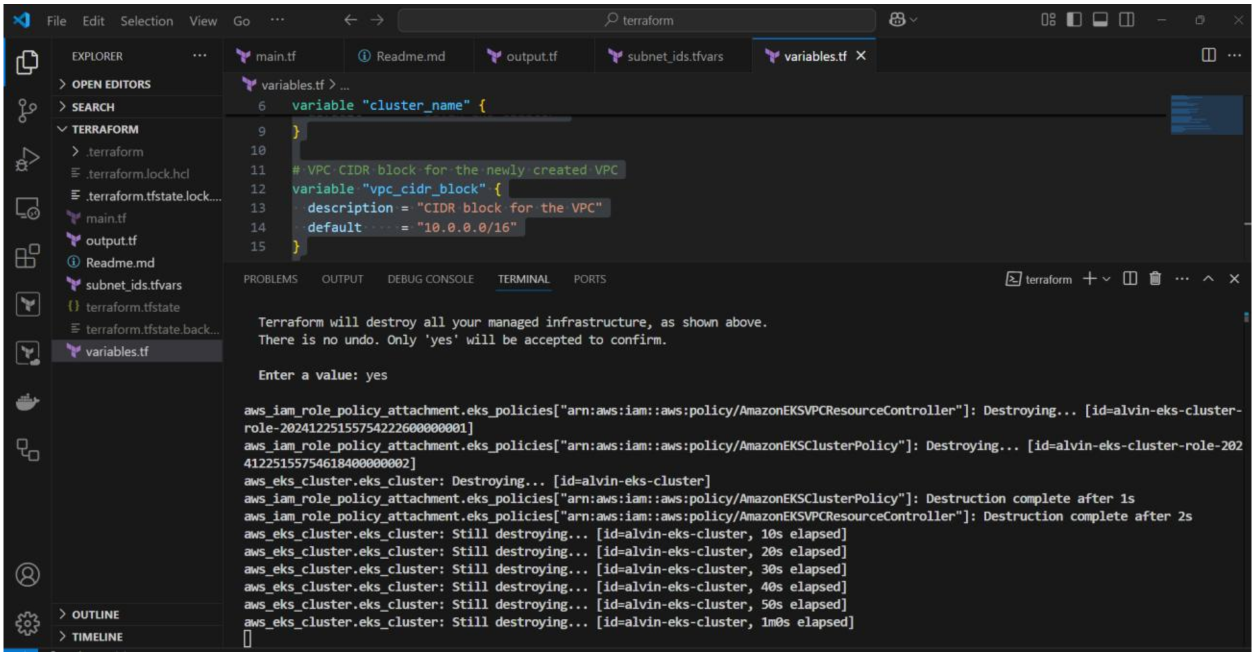
1. Run the following command:

```
terraform destroy
```



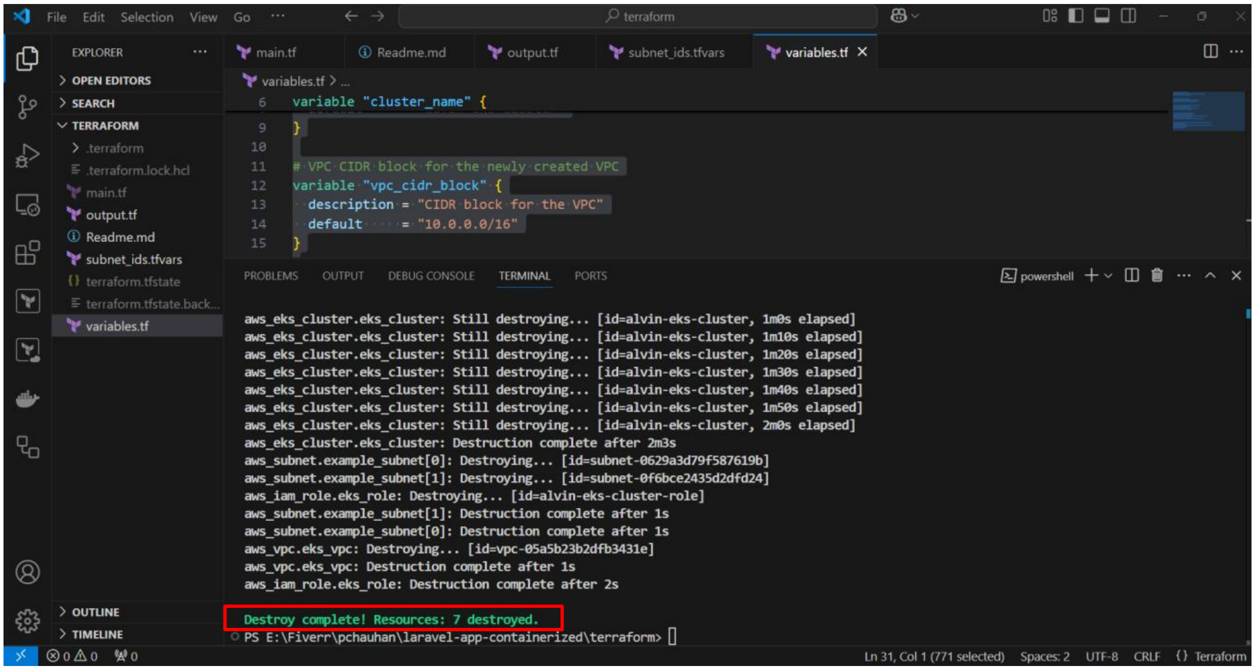
```
variable "cluster_name" {
}

# VPC CIDR block for the newly created VPC
variable "vpc_cidr_block" {
  description = "CIDR block for the VPC"
  default     = "10.0.0.0/16"
}
```

```
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform> aws configure
AWS Access Key ID [****************7E4R]:
AWS Secret Access Key [****************5xku]:
Default region name [ap-south-1]:
Default output format [json]:
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform> terraform destroy
```



```
variable "cluster_name" {
}

# VPC CIDR block for the newly created VPC
variable "vpc_cidr_block" {
  description = "CIDR block for the VPC"
  default     = "10.0.0.0/16"
}
```

```
  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

aws_iam_role_policy_attachment.eks_policies["arn:aws:iam::aws:policy/AmazonEKSVPCResourceController"]: Destroying... [id=alvin-eks-cluster-role-20241225155754222600000001]
aws_iam_role_policy_attachment.eks_policies["arn:aws:iam::aws:policy/AmazonEKSClusterPolicy"]: Destroying... [id=alvin-eks-cluster-role-20241225155754618400000002]
aws_eks_cluster.eks_cluster: Destroying... [id=alvin-eks-cluster]
aws_iam_role_policy_attachment.eks_policies["arn:aws:iam::aws:policy/AmazonEKSClusterPolicy"]: Destruction complete after 1s
aws_iam_role_policy_attachment.eks_policies["arn:aws:iam::aws:policy/AmazonEKSVPCResourceController"]: Destruction complete after 2s
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 10s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 20s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 30s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 40s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 50s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 1m0s elapsed]
```



```
variable "cluster_name" {
}

# VPC CIDR block for the newly created VPC
variable "vpc_cidr_block" {
  description = "CIDR block for the VPC"
  default     = "10.0.0.0/16"
}
```

```
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 1m0s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 1m10s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 1m20s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 1m30s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 1m40s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 1m50s elapsed]
aws_eks_cluster.eks_cluster: Still destroying... [id=alvin-eks-cluster, 2m0s elapsed]
aws_eks_cluster.eks_cluster: Destruction complete after 2m3s
aws_subnet.example_subnet[0]: Destroying... [id=subnet-0629a3d79f587619b]
aws_subnet.example_subnet[1]: Destroying... [id=subnet-0f6bce2435d2dfd24]
aws_iam_role.eks_role: Destroying... [id=alvin-eks-cluster-role]
aws_subnet.example_subnet[1]: Destruction complete after 1s
aws_subnet.example_subnet[0]: Destruction complete after 1s
aws_vpc.eks_vpc: Destroying... [id=vpc-05a5b23b2dfb3431e]
aws_vpc.eks_vpc: Destruction complete after 1s
aws_iam_role.eks_role: Destruction complete after 2s

Destroy complete! Resources: 7 destroyed.
PS E:\Fiverr\pchauhan\laravel-app-containerized\terraform>
```

Conclusion

This configuration demonstrates a complete setup for deploying a scalable and secure EKS cluster on AWS. It’s designed for flexibility, allowing easy modifications to suit specific requirements. If you're interested in modern DevOps practices or cloud-native solutions, this is a great starting point!