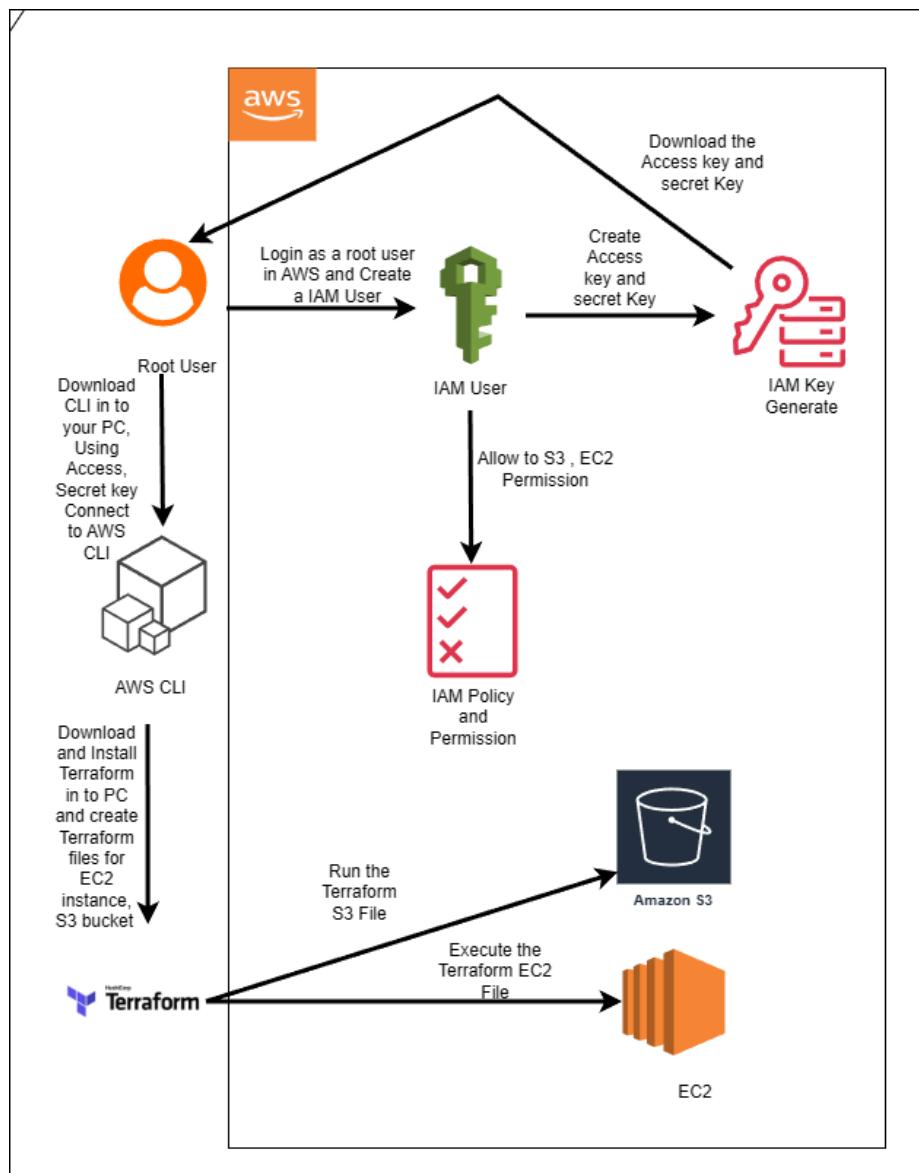


Step-by-step guide on how to create an EC2 instance and S3 bucket using Terraform

Terraform is a tool for building, changing, and versioning infrastructure safely and efficiently. It takes your infrastructure you have defined in code and makes it real! The beauty of what Terraform does is that it does not ask you how to get from the infrastructure you have to the infrastructure you want, it just asks you what you want the world to look like and then it does the hard work.



- ❖ **Step 1: Download the Terraform** in to your PC. Install Chocolatey recommended for windows, Open power shell as administrator then paste the below command.

- ❖ First install Chocolatey (if not already installed),

```
Set-ExecutionPolicy Bypass -Scope Process -Force
[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072
iex ((New-Object
System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
```

- ❖ Install Terraform

choco install terraform

- ❖ Option 2: Go to the official Terraform website, select your operating system (Windows or macOS), choose the version you need, and download the package

<https://developer.hashicorp.com/terraform/install>

After Installation done go to PowerShell or MacOs terminal just type below command then your Terraform version show like that,

terraform --version

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/powershell

PS C:\Windows\system32> # Run PowerShell as Administrator
>> Set-ExecutionPolicy Bypass -Scope Process -Force
>> [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072
>> iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
WARNING: 'choco' was found at 'C:\ProgramData\chocolatey\bin\choco.exe'.
WARNING: An existing Chocolatey installation was detected. Installation will not continue. This script will not
overwrite existing installations.
If there is no Chocolatey installation at 'C:\ProgramData\chocolatey', delete the folder and attempt the installation
again.

Please use choco upgrade chocolatey to handle upgrades of Chocolatey itself.
If the existing installation is not functional or a prior installation did not complete, follow these steps:
- Backup the files at the path listed above so you can restore your previous installation if needed.
- Remove the existing installation manually.
- Rerun this installation script.
- Reinstall any packages previously installed, if needed (refer to the lib folder in the backup).

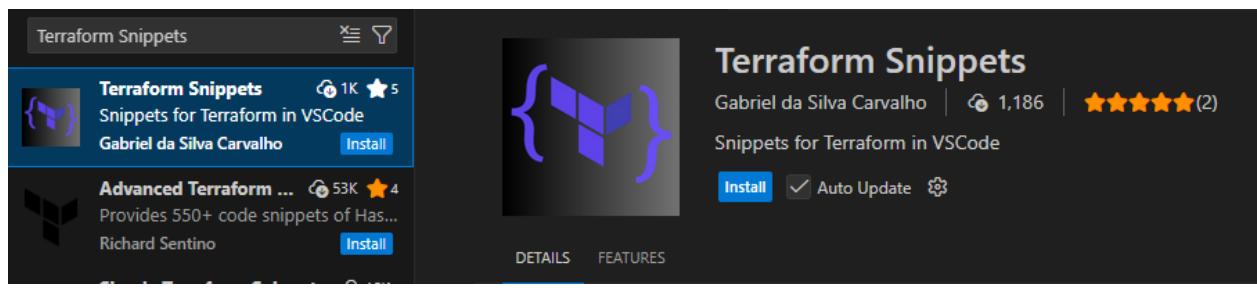
Once installation is completed, the backup folder is no longer needed and can be deleted.
PS C:\Windows\system32> choco install terraform -y
Chocolatey v0.10.15
Installing the following packages:
terrafrom
By installing you accept licenses for the packages.
Progress: Downloading terraform 1.14.3... 100%
terrafrom v1.14.3 [Approved]
terrafrom package files install completed. Performing other installation steps.
Removing old terraform plugins
Downloading terraform 64 bit
  from 'https://releases.hashicorp.com/terrafrom/1.14.3/terrafrom_1.14.3_windows_amd64.zip'
Progress: 100% - Completed download of C:\Users\Nethum\AppData\Local\Temp\chocolatey\terrafrom\1.14.3\terrafrom_1.14.3_windows_amd64.zip (29.8 MB).
Download of terraform_1.14.3_windows_amd64.zip (29.8 MB) completed.
Hashes match.
Extracting C:\Users\Nethum\AppData\Local\Temp\chocolatey\terrafrom\1.14.3\terrafrom_1.14.3_windows_amd64.zip to C:\ProgramData\chocolatey\lib\terrafrom\tools...
C:\ProgramData\chocolatey\lib\terrafrom\tools
'ShimGen has successfully created a shim for terraform.exe
The install of terraform was successful.
Software installed to 'C:\ProgramData\chocolatey\lib\terrafrom\tools'

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\Windows\system32> terraform --version
Terraform v1.14.3
on windows_amd64
PS C:\Windows\system32>
```

- ❖ **Step 2: Open VS code** and go to extensions then download supportive tools for properly run Terraform file,
 - ❖ Install Hashicope Terraform



- ❖ Install Terraform Snippets,



- ❖ **Step 3: Download AWS CLI** : Go to the official AWS website, download the latest version to your PC, and verify the download was successful.

<https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

check whether AWS status :- aws --version

AWS Command Line Interface User Guide for Version 2

Install and update requirements

- We support the AWS CLI on Microsoft-supported versions of 64-bit Windows.
- Admin rights to install software

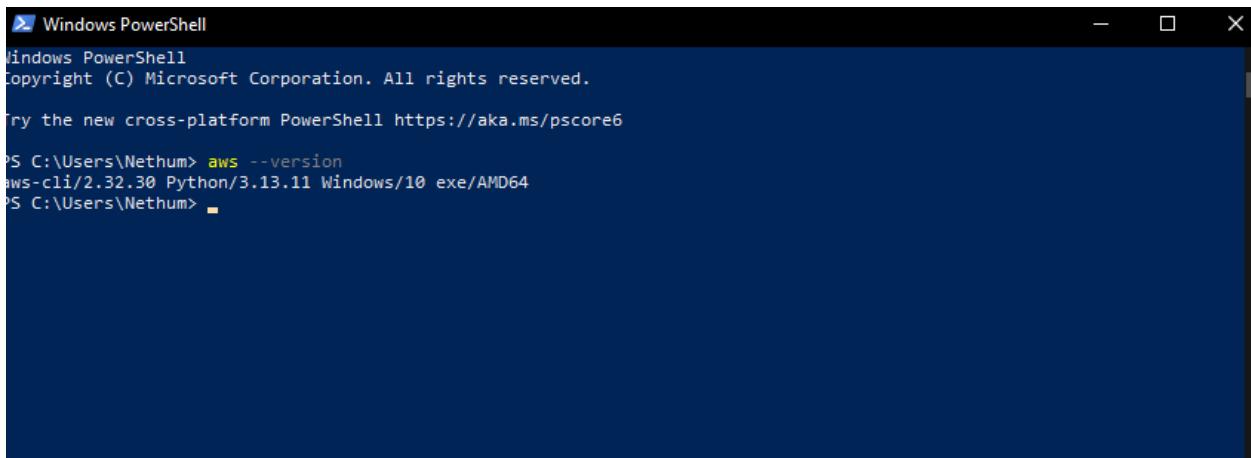
Install or update the AWS CLI

To update your current installation of AWS CLI on Windows, download a new installer each time you update to overwrite previous versions. AWS CLI is updated regularly. To see when the latest version was released, see the [AWS CLI version 2 Changelog](#) on GitHub.

- Download and run the AWS CLI MSI installer for Windows (64-bit):
<https://awscli.amazonaws.com/AWSCLIV2.msi>
 Alternatively, you can run the `msiexec` command to run the MSI installer.


For various parameters that can be used with `msiexec`, see [msiexec](#) on the Microsoft Docs website. For example, you can use the `/qn` flag for a silent

❖ AWS Status



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Nethum> aws --version
aws-cli/2.32.30 Python/3.13.11 Windows/10 exe/AMD64
PS C:\Users\Nethum>
```

❖ Open VS code and Install AWS tool kit to configure AWS service.

EXTENSIONS: MARKETPLACE

Extension: AWS Toolkit

AWS Toolkit 3.7M ⭐ 2.5
Including CodeCatalyst, Infrastructure... [Install](#)

AWS Toolbox 8K ⭐ 5
AWS Resource Visualization [Install](#)

Common Fate - AWS Toolkit 1K ⭐ 5
A fast interface to the cloud. AWS a... Common Fate [Install](#)

AWS Athena Query Tool 45
Run AWS Athena SQL queries direct... Mark Sawczuk [Install](#)

Databricks Power Tools 136K ⭐ 4.5
Run notebooks cell-by-cell, browse... Databricks [Install](#)

Stackery Serverless Toolkit 7K ⭐ 5
CloudFormation architecture editin... Stackery [Install](#)

LocalStack Toolkit 10K
LocalStack - Run locally, deploy glo... LocalStack [Install](#)

IAM Tools Extension Pack 983
A collection of tools that help you ... Ian McKay [Install](#)

Toolbox-VADB 0

AWS Toolkit
Amazon Web Services [amazon.com](#) | 3,790,665 | ⭐ 2.5 (73)
Including CodeCatalyst, Infrastructure Composer, and support for Lambda, S3, CloudWatch Logs, CloudFormation, and...

[Install](#) Auto Update

DETAILS FEATURES CHANLOG

Follow @aws YouTube 197M Installs 3.8M

Amazon Q and CodeWhisperer
Amazon CodeWhisperer is now part of Amazon Q. Try the Amazon Q extension.

Getting Started

- Open the AWS Toolkit extension
- Sign in with your AWS credentials

EXPLORER index.js

```
calculator > src > pages > # index.js > ...
1 // This file contains the list of pages that will be rendered in the StartPage component
2 import CalculatorPage from './calculatorPage';
3 import VolumeCalculatorPage from './volumeCalculatorPage';
4
5 // pages and the navigation bar. Each page has a title, description, image, path, and
6 // URL. Each element from the array will be rendered in the pages.
7
8 const pages = [
9   {
10     title: 'Calculator',
11     description: 'Provides a simple calculator for integers. Supports +, -, *, / operations',
12     path: '/calculator',
13     image: 'calculatorIcon.png',
14   },
15 ]
```

Marketplace Identifier: amazonwebservices.aws-toolkit-vscode
Version: 3.91.0
Published: 6 years ago
Last Released: 2 weeks ago

Categories: Linters, Debuggers, Visualization, Notebooks

Resources: Repository, Issues, License, Amazon Web Services, Marketplace

Finish Setup

- ❖ **Step 4 Create an IAM user.** Under Permissions policies, attach the policies AmazonS3FullAccess and AmazonEC2FullAccess. Then, go to Security credentials, select Create access key, and generate an access key ID and a secret access key.

❖ IAM User Creation

Specify user details

User details

User name
Terraform-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*
In addition to console access, users with SigninLocalDevelopmentAccess permissions can use the same console credentials for programmatic access without the need for access keys.

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](#)

Cancel **Next**

❖ Add permission and policy in to IAM user,

Review

The following policies will be attached to this user. [Learn more](#)

User details

User name
Terraform-user

Permissions summary (2)

Name	Type	Used as
AmazonS3FullAccess	AWS managed	Permissions policy
AmazonEC2FullAccess	AWS managed	Permissions policy

Add permissions

Cancel **Previous** **Add permissions**

❖ Access Key Creation,

The screenshot shows the AWS IAM 'Create access key' process at Step 2 - optional. A green banner at the top indicates '2 policies added to Terraform-user'. The main content area is titled 'Access key best practices & alternatives' with a link to 'Info'. It advises against using long-term credentials like access keys to improve security. Below this, a 'Use case' section is shown with three options: 'Command Line Interface (CLI)' (selected), 'Local code', and 'Application running on an AWS compute service'. At the bottom of the page are links for 'CloudShell', 'Feedback', 'Console Mobile App', and standard footer links for 'Privacy', 'Terms', and 'Cookie preferences'.

❖ Provide tag name for the Access Key creation,

The screenshot shows the AWS IAM 'Create access key' process at Step 2 - optional. A green banner at the top indicates '2 policies added to Terraform-user'. The main content area is titled 'Set description tag - optional' with a link to 'Info'. It explains that the description will be attached to the user as a tag. A 'Description tag value' input field contains the text 'Terraform-access'. Below the input field, a note states 'Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @'. At the bottom of the page are 'Cancel', 'Previous', and 'Create access key' buttons. The footer includes links for 'CloudShell', 'Feedback', 'Console Mobile App', and standard footer links for 'Privacy', 'Terms', and 'Cookie preferences'.

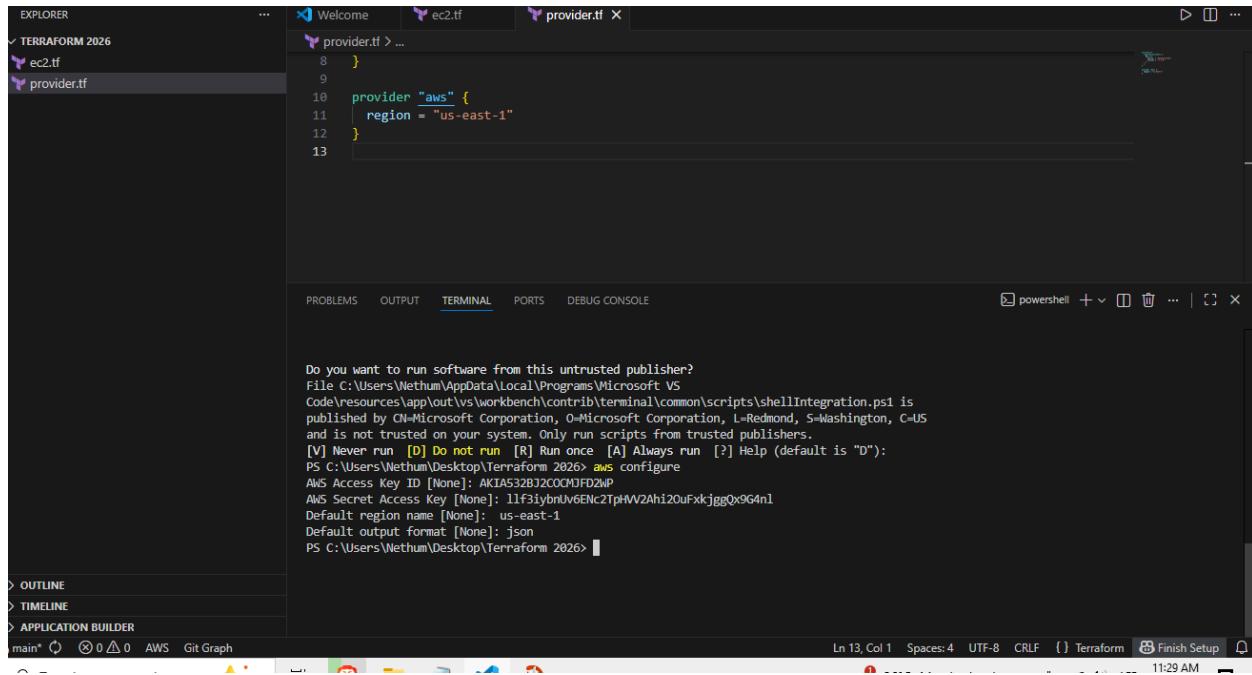
- ❖ Generated the Access key and secret key,

This screenshot shows the 'Create access key' step in the AWS IAM console. The user 'Terraform-user' is selected. A green banner at the top states: '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.' The process is divided into three steps: Step 1 (Access key best practices & alternatives), Step 2 - optional (Set description tag), and Step 3 (Retrieve access keys). Step 3 is currently active. The 'Access key' section shows the access key ID 'AKIA532BJ2COCMJFD2WP' and a redacted secret access key. Below this, the 'Access key best practices' section lists several guidelines.

- ❖ Step 5: Create a Terraform file in VS Code to define an EC2 instance. Provide the required AMI ID from the AWS Management Console's AMI Catalog.

This screenshot shows the AWS Management Console's AMI Catalog search results for 'ubuntu'. The search bar at the top contains 'ubuntu'. The results are categorized into 'Quick Start AMIs (8)', 'My AMIs (0)', 'AWS Marketplace AMIs (1346)', and 'Community AMIs (500)'. The 'Quick Start AMIs (8)' tab is selected. The results are further refined by filters like 'Free tier only', 'OS category', and 'Architecture'. The first result is 'ubuntu' (Ubuntu Server 24.04 LTS (HVM), SSD Volume Type), which is a 'Verified provider'. The 'Select' button is visible next to the '64-bit (x86)' option.

- ❖ **Step 6: Configure AWS CLI:** Open your terminal and configure the AWS CLI by providing your IAM user's access key, secret key, region, and output format. Then, create separate Terraform files for an S3 bucket and an EC2 instance,



The screenshot shows the VS Code interface with the 'TERRAFORM 2026' workspace open. The 'EXPLORER' sidebar shows files: provider.tf, ec2.tf, and ec2.tf. The 'TERMINAL' tab is active, displaying the configuration of the AWS provider in provider.tf:

```

provider "aws" {
  region = "us-east-1"
}

```

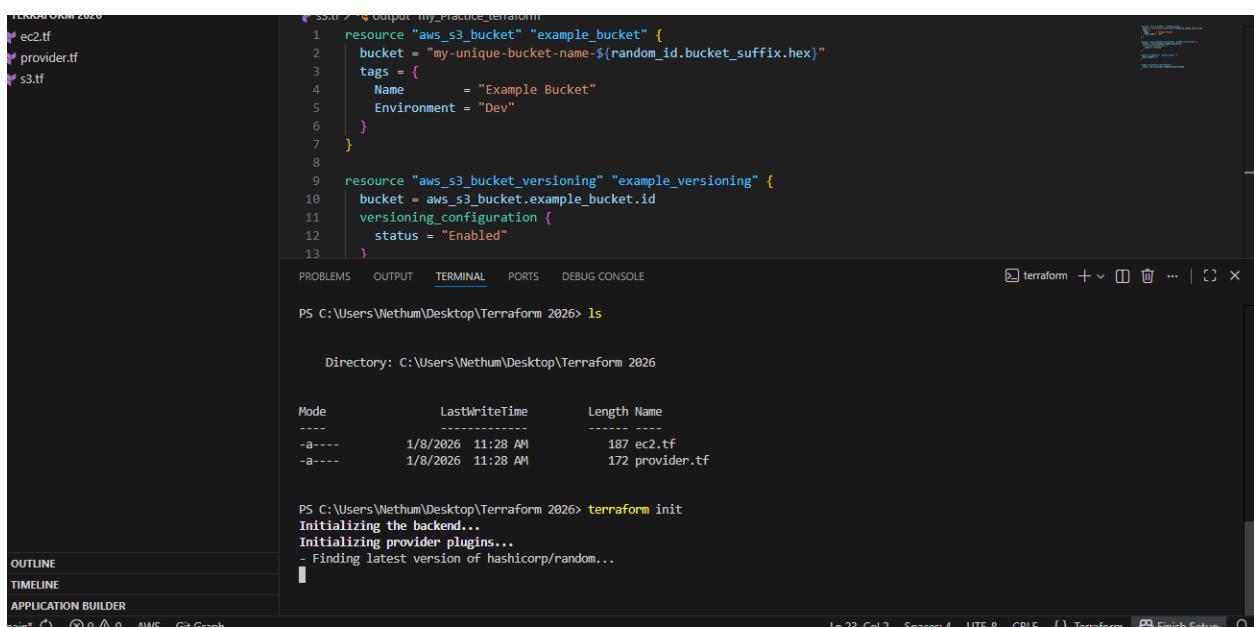
Below the terminal, a PowerShell window shows the execution of the 'aws configure' command:

```

Do you want to run software from this untrusted publisher?
File C:\Users\Nethum\AppData\Local\Programs\Microsoft VS Code\resources\app\out\vs-workbench\contrib\terminal\common\scripts\shellIntegration.ps1 is published by CN-Microsoft Corporation, O-Microsoft Corporation, L-Redmond, S-Washington, C-US and is not trusted on your system. Only run scripts from trusted publishers.
[V] Never run [D] Do not run [R] Run once [A] Always run [?] Help (default is "D"):
PS C:\Users\Nethum\Desktop\Terraform 2026> aws configure
AWS Access Key ID [None]: AKIA52B3J2C0CMFD2kP
AWS Secret Access Key [None]: 1lf3t1ybnlVgEnc2TpHW2Ah120uFxkjggQx9G4n1
Default region name [None]: us-east-1
Default output format [None]: json
PS C:\Users\Nethum\Desktop\Terraform 2026>

```

- After Configure AWS CLI and create Terraform files, Finally Execute the terraform Commands,
- Command : `terraform init`



The screenshot shows the VS Code interface with the 'TERRAFORM 2026' workspace open. The 'EXPLORER' sidebar shows files: ec2.tf, provider.tf, and s3.tf. The 'TERMINAL' tab is active, displaying the execution of the 'terraform init' command:

```

PS C:\Users\Nethum\Desktop\Terraform 2026> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/random...

```

- Command :- `terraform plan`

```
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\Nethum\Desktop\Terraform 2026> terraform plan

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_instance.ubuntu_server will be created
+ resource "aws_instance" "ubuntu_server" {
    + ami                                = "ami-053b0d53c279acc90"
    + 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)

Ln 23, Col 2  Spaces: 4  UTF-8  CRLF  { } Terraform  Finish Setup  11:38 AM
```

- Command : terraform apply

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER**: Shows a tree view of the project structure under "TERRAFORM 2026". The "ec2.tf" file is currently selected.
- TERMINAL**: Displays the output of the Terraform command, showing the creation of resources like a bucket and an instance.
- OUTPUT**: Shows the message "Apply complete! Resources: 4 added, 0 changed, 0 destroyed."
- PROBLEMS**: Shows no errors or warnings.
- PORTS**: Shows no active ports.
- DEBUG CONSOLE**: Shows the command "PS C:\Users\Wethum\Desktop\Terraform_2026>".

Code content from ec2.tf:

```
provider "aws" {
  region = "us-east-1"
}

resource "aws_instance" "ubuntu_server" {
  ami           = "ami-0885b1f6bd170450c"
  instance_type = "t3.micro" # Changed from t2.micro to t3.micro

  tags = [
    { Name = "ubuntu-free-tier-instance" }
  ]
}
```

Terminal Output:

```
Enter a value: yes

random_id.bucket_suffix: Creating...
random_id.bucket_suffix: Creation complete after 0s [id=b0xzGg]
aws_s3_bucket.example_bucket: Creating...
aws_instance.ubuntu_server: Creating...
aws_instance.ubuntu_server: Still creating... [0m01s elapsed]
aws_s3_bucket.example_bucket: Still creating... [0m01s elapsed]
aws_s3_bucket.example_bucket: Creation complete after 11s [id=my-unique-bucket-name-6c3c591a]
aws_s3_bucket_versioning.example_versioning: Creating...
aws_s3_bucket_versioning.example_versioning: Creation complete after 2s [id=my-unique-bucket-name-6c3c591a]
aws_instance.ubuntu_server: Creation complete after 17s [id=i-0c011952bce448f84]

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

Outputs:

my_Practice_terraform = "my-unique-bucket-name-6c3c591a"
```

- After executing the Terraform commands, go to the AWS Management Console to verify whether the EC2 instance and S3 bucket were created successfully.

- EC2 Instance Status,

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with options like Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager (New), and Images. The main content area shows a table titled "Instances (1/1) Info" with one row. The row details an instance named "ubuntu-free-ti..." with Instance ID "i-0c011952bce448f84". The instance is listed as "Running" with status "t3.micro", "Initializing", and "View alarms" for "us-east-1c". Below the table, a detailed view for "i-0c011952bce448f84 (ubuntu-free-tier-instance)" is shown with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. Under "Instance summary", it shows the Instance ID, Public IPv4 address (34.227.228.126), Private IPv4 addresses (172.31.22.196), and Public DNS.

- S3 bucket Status

The screenshot shows the AWS Amazon S3 General purpose buckets page. The left sidebar includes Buckets (General purpose buckets, Directory buckets, Table buckets, Vector buckets), Access management and security (Access Points, Access Points for FSx, Access Grants, IAM Access Analyzer), and Storage management and insights (Storage Lens, Batch Operations). The main content shows a table for "General purpose buckets (1) Info" with one entry: "my-unique-bucket-name-6c3c591a" in US East (N. Virginia) us-east-1, created on January 8, 2026, at 12:15:32 (UTC+05:30). There are also sections for "Account snapshot" (View dashboard, Updated daily) and "External access summary - new" (Info, Updated daily).

Terraform File Structure

Ec2.tf

```
provider "aws" {
  region = "us-east-1"
}

resource "aws_instance" "ubuntu_server" {
  ami           = "ami-0885b1f6bd170450c"
  instance_type = "t3.micro"  # Changed from t2.micro to t3.micro

  tags = {
    Name = "ubuntu-free-tier-instance"
  }
}
```

S3.tf

```
resource "aws_s3_bucket" "example_bucket" {
  bucket = "my-unique-bucket-name-${random_id.bucket_suffix.hex}"
  tags = {
    Name      = "Example Bucket"
    Environment = "Dev"
  }
}

resource "aws_s3_bucket_versioning" "example_versioning" {
  bucket = aws_s3_bucket.example_bucket.id
  versioning_configuration {
    status = "Enabled"
  }
}

resource "random_id" "bucket_suffix" {
  byte_length = 4
}

output "my_Practice_terraform" {
  value = aws_s3_bucket.example_bucket.bucket
}
```