

## Pre-Requisites:

Git bash, node, npm, docker desktop already installed

## TASK#1:

*Task 1: Get Started with Terraform and AWS EC2*

*Goal: Set up Terraform and provision a basic EC2 instance.*

*Why: This helps you understand the core Terraform workflow and the AWS provider.*

*What You'll Learn: Installing Terraform, configuring AWS CLI, writing your first .tf file, and connecting to EC2.*

- *Install Terraform on your local machine.*
- *Set up AWS CLI with IAM credentials (learn about access keys).*
- *Create a Terraform script to launch a t2.micro EC2 instance.*
- *Use Terraform outputs to display the public IP.*
- *SSH into the instance and install Nginx to verify connectivity.*

installing aws cli

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

AWS Command Line Interface User Guide for Version 2

Windows

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.

```
C:\> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi
```

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 installation.

```
C:\> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi /qn
```

On this page

AWS CLI install and update instructions

Troubleshooting AWS CLI install and uninstall errors

Next steps

Recommended tasks

How to

Verify Session Manager plugin installation

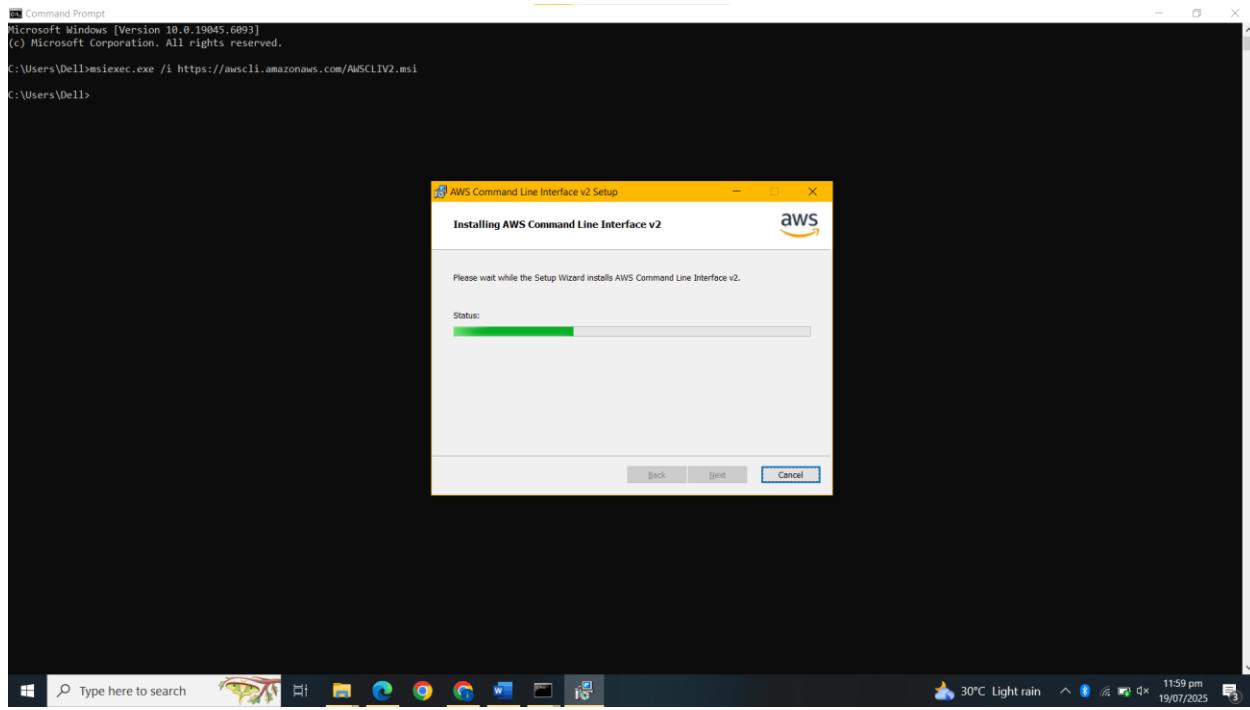
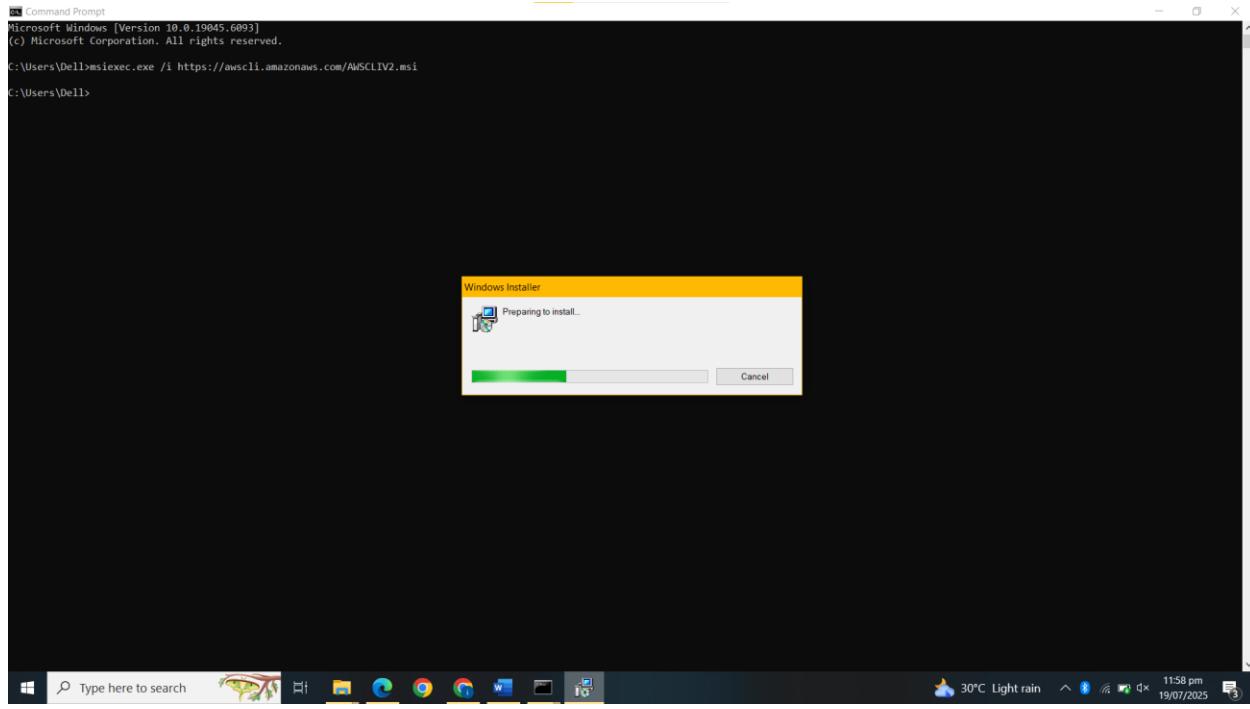
Learn about

Supported AWS Regions for CloudShell

Did this page help you?

Yes No

Provide feedback



## Download terraform

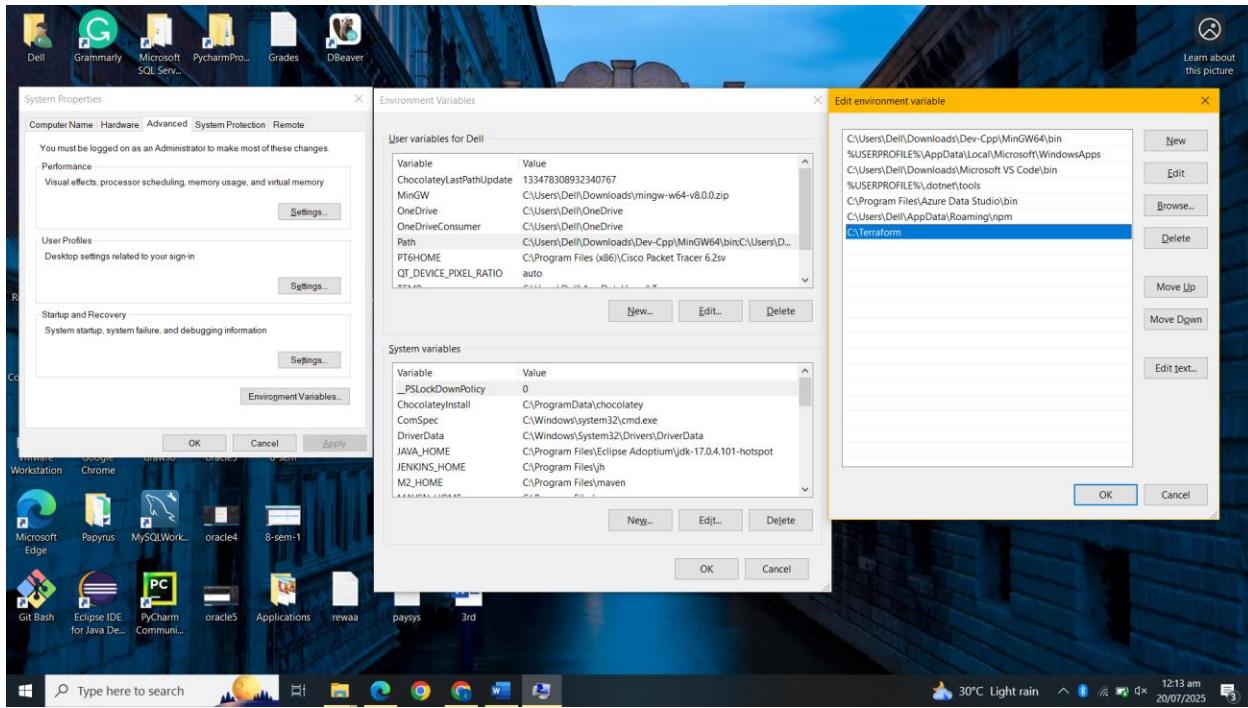
<https://developer.hashicorp.com/terraform/install>

The screenshot shows the Terraform installation page. The left sidebar has sections for 'Terraform Home', 'Install Terraform' (selected), 'Operating Systems' (macOS selected), 'Windows', 'Linux', 'FreeBSD', 'OpenBSD', 'Solaris', 'Release information', 'Next steps', and 'Resources'. The main content area has two sections: 'macOS' and 'Windows'. The 'macOS' section has a 'Package manager' section with brew commands and binary download links for 'AMD64' (Version: 1.12.2) and 'ARM64' (Version: 1.12.2). The 'Windows' section has a 'Binary download' section with links for '386' (Version: 1.12.2) and 'AMD64' (Version: 1.12.2). On the right, there's an 'About Terraform' summary, 'Featured docs' (Introduction to Terraform, Configuration Language, Terraform CLI, HCP Terraform, Provider Use), and an 'HCP Terraform' section.

Extract the files and put it in terraform folder like this in the c drive:

The screenshot shows a Windows File Explorer window with the path 'This PC > Local Disk (C:) > Terraform'. Inside the 'Terraform' folder, there are two items: 'LICENSE' (Text Document, 5 KB) and 'terraform' (Application, 93,519 KB). The left sidebar shows standard Windows navigation options like Quick access, Desktop, Downloads, and OneDrive.

Add it in the environment variables in the path



Ensure version of everything on git bash

```

MINGW64 /c/Users/Dell
$ node -v
v20.10.0

$ npm -v
10.2.3

$ docker --version
Docker version 26.0.0, build 2ae903e

$ aws --version
aws-cli/2.27.55 Python/3.13.4 Windows/10 exe/AMD64

$ terraform -v
Terraform v1.12.2
on windows_amd64

```

Create an AWS account and sign it:

<https://aws.amazon.com/>

## Create IAM User

The screenshot shows the 'Specify user details' step of the IAM user creation wizard. The user name is set to 'terraform-user'. A note indicates that the user name can have up to 64 characters and must start with a letter. An optional checkbox for console access is present. A callout box highlights the option to generate programmatic access keys after user creation.

Step 1  
Specify user details  
Step 2  
Set permissions  
Step 3  
Review and create

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*  
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](#)

Cancel Next

The screenshot shows the 'Set permissions' step of the IAM user creation wizard. It allows adding users to groups or attaching policies directly. The 'Attach policies directly' option is selected. A note about using groups for managing permissions by job function is present. The 'Permissions policies' section shows a search bar for filtering policies by name, type, and attached entities. One policy, 'AmazonEC2FullAccess', is listed under 'AWS managed' policies.

Step 1  
Specify user details  
Step 2  
Set permissions  
Step 3  
Review and create

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 (1/1378)  
Choose one or more policies to attach to your new user.

Filter by Type

Policy name	Type	Attached entities
AmazonEC2FullAccess	AWS managed	0

▶ Set permissions boundary - *optional*

Cancel Previous Next

Screenshot of the AWS IAM User Creation Step 2: Set permissions screen.

The URL is [us-east-1.console.aws.amazon.com/iam/home?region=eu-north-1#/users/create](https://us-east-1.console.aws.amazon.com/iam/home?region=eu-north-1#/users/create).

**Permissions options:**

- Add user to group
- Copy permissions
- Attach policies directly

**Permissions policies (2/1378):**

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

Filter by Type	
Policy name	Type
AmazonVPCF	AWS managed

**Set permissions boundary - optional**

**User details:**

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

**Permissions summary:**

Name	Type	Used as
AmazonEC2FullAccess	AWS managed	Permissions policy
AmazonVPCFullAccess	AWS managed	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.

**Create user**

The screenshot shows the AWS IAM User Details page for a user named 'terraform-user'. The left sidebar includes links for Identity and Access Management (IAM), Access management (Users, Roles, Policies, Identity providers, Account settings, Root access management), and Access reports (Resource Analyzer, Unused access, Analyzer settings). The main content area displays the 'Summary' tab for 'terraform-user', showing the ARN (arn:aws:iam:400375465965:user/terraform-user), creation date (July 20, 2025, 12:49 (UTC+05:00)), and last console sign-in (not applicable). It also lists 'Access key 1' and a 'Create access key' button. Below the summary, tabs for 'Permissions', 'Groups', 'Tags', 'Security credentials' (selected), and 'Last Accessed' are visible. The 'Security credentials' section contains 'Console sign-in' information (Console sign-in link: https://400375465965.signin.aws.amazon.com/console, Console password: Not enabled) and a 'Multi-factor authentication (MFA)' section (0 MFA devices assigned). A 'Create access key' button is located at the bottom of this section. The status bar at the bottom right shows 'Match' and system icons.

The screenshot shows the 'Create access key' page for the 'terraform-user' user. The left sidebar shows the user's details. The main content area is titled 'Access key best practices & alternatives' (info). It provides guidance on avoiding long-term credentials like access keys to improve security. Below this, a 'Use case' section lists several options: 'Command Line Interface (CLI)' (selected), 'Local code', 'Application running on an AWS compute service', 'Third-party service', 'Application running outside AWS', and 'Other'. Each option has a brief description. The status bar at the bottom right shows 'Humid' and system icons.

The screenshot shows the AWS IAM 'Create access key' wizard. The current step is 'Set description tag - optional'. A description tag value 'Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.' is entered. The user has selected the 'Set description tag' option. The wizard includes 'Cancel', 'Previous', and 'Create access key' buttons.

The screenshot shows a Windows terminal window with the command `aws sts get-caller-identity` running. The output displays the user ID, account number, and ARN of the user. The terminal window is titled 'MINGW64' and is located at 'C:\Users\user'. The system tray shows the date and time as 20/07/2025 and 12:51 pm.

**no error, it means your AWS CLI is now connected to your account!**

AWS CLI helps Terraform communicate with your AWS account using your IAM credentials.

**Key pair**  
A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

**Name**  
terraform-key  
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

**Key pair type** | **Info**  
 RSA       ED25519

**Private key file format**  
 .pem  
For use with OpenSSH  
 .ppk  
For use with PuTTY

**Tags - optional**  
No tags associated with the resource.  
[Add new tag](#)  
You can add up to 50 more tags.

[Create key pair](#)

This .pem file allows you to **SSH (connect)** to the EC2 instance.

Name	Date modified	Type	Size
main.tf	20/07/2025 2:02 pm	TF File	1 KB
terraform-key	20/07/2025 2:07 pm	PEM File	2 KB

Use the accurate ami in the main.tf  
checking with terraform plan to see a dry run, that is what will be applied

```
bell@DESKTOP-VVSNUS0 MINGW64 /d/Freelancing/Uwork/Task-2/Task-1-EC2-NGINX
$ ls
main.tf  terraform.tfstate  terraform-key.pem

bell@DESKTOP-VVSNUS0 MINGW64 /d/Freelancing/Uwork/Task-2/Task-1-EC2-NGINX
$ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.4.0

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.

If you ever set or change modules or backend configuration for Terraform,
run this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

bell@DESKTOP-VVSNUS0 MINGW64 /d/Freelancing/Uwork/Task-2/Task-1-EC2-NGINX
$ terraform validate
Success! The configuration is valid.

bell@DESKTOP-VVSNUS0 MINGW64 /d/Freelancing/Uwork/Task-2/Task-1-EC2-NGINX
$ 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.nginx_server will be created
+ resource "aws_instance" "nginx_server" {
    ami                               = "ami-050fd9796aa387c0d"
    arm                             = (known after apply)
    associate_public_ip_address      = (known after apply)
    availability_zone                = (known after apply)
    disable_api_stop                 = (known after apply)
    disable_api_termination          = (known after apply)
    ebs_optimized                   = (known after apply)
    enable_primary_ipv6              = (known after apply)
    get_password_data               = false
    host_id                          = (known after apply)
    host_resource_group_arn          = (known after apply)
    iam_instance_profile             = (known after apply)
    id                               = (known after apply)

    + ami
    + arm
    + associate_public_ip_address
    + availability_zone
    + disable_api_stop
    + disable_api_termination
    + ebs_optimized
    + enable_primary_ipv6
    + get_password_data
    + host_id
    + host_resource_group_arn
    + iam_instance_profile
    + id
}
```

```
bell@DESKTOP-VVSNUS0 MINGW64 /d/Freelancing/Uwork/Task-2/Task-1-EC2-NGINX
+ metadata_options (known after apply)
+ network_interface (known after apply)
+ private_dns_name_options (known after apply)
+ root_block_device (known after apply)
}

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

Changes to Outputs:
+ public_ip = (known after apply)

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.

bell@DESKTOP-VVSNUS0 MINGW64 /d/Freelancing/Uwork/Task-2/Task-1-EC2-NGINX
$ terraform apply

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.nginx_server will be created
+ resource "aws_instance" "nginx_server" {
    ami                               = "ami-050fd9796aa387c0d"
    arm                             = (known after apply)
    associate_public_ip_address      = (known after apply)
    availability_zone                = (known after apply)
    disable_api_stop                 = (known after apply)
    disable_api_termination          = (known after apply)
    ebs_optimized                   = (known after apply)
    enable_primary_ipv6              = (known after apply)
    get_password_data               = false
    host_id                          = (known after apply)
    host_resource_group_arn          = (known after apply)
    iam_instance_profile             = (known after apply)
    id                               = (known after apply)
    instance_initiated_shutdown_behavior = (known after apply)
    instance_lifecycle               = (known after apply)
    instance_state                  = (known after apply)
    instance_type                   = "t3.micro"
    ipv6_address_count              = (known after apply)
    ipv6_addresses                  = "terraform-key"
    key_name                        = (known after apply)
    monitoring                      = (known after apply)
    outpost_arn                     = (known after apply)

    + ami
    + arm
    + associate_public_ip_address
    + availability_zone
    + disable_api_stop
    + disable_api_termination
    + ebs_optimized
    + enable_primary_ipv6
    + get_password_data
    + host_id
    + host_resource_group_arn
    + iam_instance_profile
    + id
    + instance_initiated_shutdown_behavior
    + instance_lifecycle
    + instance_state
    + instance_type
    + ipv6_address_count
    + ipv6_addresses
    + key_name
    + monitoring
    + outpost_arn
}
```

```

MINGW64/d/Freelancing/Uptwork/Task-2/Task-1-EC2-NGINX
+ user_data_base64          = (known after apply)
+ user_data_replace_on_change = false
+ vpc_security_group_ids     = (known after apply)

+ capacity_reservation_specification (known after apply)

+ cpu_options (known after apply)

+ ebs_block_device (known after apply)

+ enclave_options (known after apply)

+ ephemeral_block_device (known after apply)

+ instance_market_options (known after apply)

+ maintenance_options (known after apply)

+ metadata_options (known after apply)

+ network_interface (known after apply)

+ private_dns_name_options (known after apply)

+ root_block_device (known after apply)
}

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

Changes to Outputs:
+ public_ip = (known after apply)

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.nginx_server: Creating...
aws_instance.nginx_server: Still creating... [00m10s elapsed]
aws_instance.nginx_server: creation complete after 19s [id=i-08057ab819a682264]

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

Outputs:

public_ip = "98.80.13.12"

bell@DESKTOP-VV5NU8 MINGW64 /d/Freelancing/Uptwork/Task-2/Task-1-EC2-NGINX
$ -

```

```

MINGW64/d/Freelancing/Uptwork/Task-2/Task-1-EC2-NGINX
+ prefix_list_ids  = []
+ protocol        = "tcp"
+ security_groups = []
+ self             = false
+ to_port          = 80
  # (1 unchanged attribute hidden)
},
]
+
+ name           = "nginx-security-group"
+ name_prefix    = (known after apply)
+ owner_id       = (known after apply)
+ region         = "east-1"
+ revoke_rules_on_delete = false
+ tags_all       = (known after apply)
+ vpc_id          = (known after apply)
}

Plan: 2 to add, 0 to change, 1 to destroy.

Changes to Outputs:
- public_ip = "98.80.13.12" -> (known after apply)

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.nginx_server: Destroying... [id=i-08057ab819a682264]
aws_instance.nginx_server: Still destroying... [id=i-08057ab819a682264, 00m10s elapsed]
aws_instance.nginx_server: Still destroying... [id=i-08057ab819a682264, 00m20s elapsed]
aws_instance.nginx_server: Still destroying... [id=i-08057ab819a682264, 00m30s elapsed]
aws_instance.nginx_server: Still destroying... [id=i-08057ab819a682264, 00m40s elapsed]
aws_instance.nginx_server: Still destroying... [id=i-08057ab819a682264, 00m50s elapsed]
aws_instance.nginx_server: Destruction complete after 58s

aws_security_group.nginx_sg: Creating...
aws_security_group.nginx_sg: Creation complete after 7s [id=sg-0890473ca7b7e125d]
aws_instance.nginx_server: Creating...
aws_instance.nginx_server: Still creating... [00m10s elapsed]
aws_instance.nginx_server: Still creating... [00m20s elapsed]
aws_instance.nginx_server: creation complete after 26s [id=i-0026fb0c5d017d88c]

Apply complete! Resources: 2 added, 0 changed, 1 destroyed.

Outputs:

public_ip = "184.72.122.145"

bell@DESKTOP-VV5NU8 MINGW64 /d/Freelancing/Uptwork/Task-2/Task-1-EC2-NGINX
$ -

```

Access via web, use http as didn't do the ssl stuff for https



## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](http://nginx.org).  
Commercial support is available at [nginx.com](http://nginx.com).

*Thank you for using nginx.*



Ssh success:

```
[ec2-user@ip-172-31-23-98 ~]$ public_ip = "184.72.122.145"
[ec2-user@DESKTOP-VVNUIS0 MINGW64 /d/Freelancing/lpwork/Task-2/Task-1-EC2-NGINX
$ chmod 400 terraform-key.pem

[ec2-user@DESKTOP-VVNUIS0 MINGW64 /d/Freelancing/lpwork/Task-2/Task-1-EC2-NGINX
$ ssh -i terraform-key.pem ec2-user@184.72.122.145
The authenticity of host '184.72.122.145 (184.72.122.145)' can't be established.
ED25519 key fingerprint is SHA256:PPAKhGfJSyKEHsb80M#Ccfk+VThWPQzBzaqAgzJ/b2Y.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '184.72.122.145' (ED25519) to the list of known hosts.

A newer release of "Amazon Linux" is available.
Version 2023.8.20250715:
Run "/usr/bin/dnf check-release-update" for full release and version update info
#
#          Amazon Linux 2023
#          #####
#          #####
#          #####
#          \#/
#          V/   .->
#          V.   .->
#          \_/
#          \_/
#          \_/
[ec2-user@ip-172-31-23-98 ~]$ curl http://localhost
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light-dark; }
body { width: 35em; margin: 0 auto; font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and working. Further configuration is required.</p>
<p>For online documentation and support please refer to <a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at <a href="http://nginx.com/">nginx.com</a>.</p>
<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

```
dell@DESKTOP-VVNMU50 MINGW64 /d/Freelancing/Uptwork/Task-2/Task-1-EC2-NGINX
$ terraform destroy
aws_security_group.nginx_sg: Refreshing state... [id=sg-0890473ca7b7e125d]
aws_instance.nginx_server: Refreshing state... [id=i-0026fb0c5d017d88c]

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.nginx_server will be destroyed
resource "aws_instance" "nginx_server" {
  ami                               = "ami-050fd9796aa387c0d" -> null
  arn                             = "arn:aws:ec2:us-east-1:400375465965:instance/i-0026fb0c5d017d88c" -> null
  associate_public_ip_address      = true -> null
  availability_zone                = "us-east-1b" -> null
  disable_api_stop                 = false -> null
  enable_api_termination           = false -> null
  ebs_optimized                    = false -> null
  get_password_data                = false -> null
  hibernation                      = false -> null
  id                                = "i-0026fb0c5d017d88c" -> null
  instance_initiated_shutdown_behavior = "stop" -> null
  instance_state                   = "running" -> null
  instance_type                     = "t3.micro" -> null
  ipv6_address_count               = 0 -> null
  ipv6_addresses                   = [] -> null
  key_name                          = "terraform-key" -> null
  monitoring                        = false -> null
  placement_partition_number       = 0 -> null
  private_network_interface_id    = "eni-066d4adee7edcf175" -> null
  private_dns                       = "ip-172-31-23-98.ec2.internal" -> null
  private_ip                         = "172.31.23.98" -> null
  public_dns                         = "ec2-184-72-122-145.compute-1.amazonaws.com" -> null
  public_ip                           = "184.72.122.145" -> null
  region                            = "us-east-1" -> null
  secondary_private_ips             = [] -> null
  security_groups                   = [
    "nginx-security-group",
  ] -> null
  source_dest_check                  = true -> null
  subnet_id                          = "subnet-026751ad9cc52401f" -> null
  tags                               = {
    "Name" = "nginx-instance"
  } -> null
  tags_all                           = {
    "Name" = "nginx-instance"
  } -> null
  tenancy                            = "default" -> null
}

Type here to search 20/07/2025 3:59 pm
```

```
dell@DESKTOP-VVNMU50 MINGW64 /d/Freelancing/Uptwork/Task-2/Task-1-EC2-NGINX
  },
  {
    - cidr_blocks          = [
      "0.0.0.0/0",
    ]
    - from_port            = 80
    - ipv6_cidr_blocks     = []
    - prefix_list_ids      = []
    - protocol              = "tcp"
    - security_groups       = []
    - self                  = false
    - to_port               = 80
    # (1 unchanged attribute hidden)
  },
  - name                  = "nginx-security-group" -> null
  - owner_id              = "000375465965" -> null
  - region                = "us-east-1" -> null
  - revoke_rules_on_delete = false -> null
  - tags                  = {} -> null
  - tags_all              = {} -> null
  - vpc_id                = "vpc-0699b5de9a5d4840b" -> null
  # (1 unchanged attribute hidden)
}

Plan: 0 to add, 0 to change, 2 to destroy.

Changes to Outputs:
  - public_ip = "184.72.122.145" -> null

Do you really want to destroy all resources?
  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_instance.nginx_server: Destroying... [id=i-0026fb0c5d017d88c]
aws_instance.nginx_server: Still destroying... [id=i-0026fb0c5d017d88c, 00m10s elapsed]
aws_instance.nginx_server: Still destroying... [id=i-0026fb0c5d017d88c, 00m20s elapsed]
aws_instance.nginx_server: Still destroying... [id=i-0026fb0c5d017d88c, 00m30s elapsed]
aws_instance.nginx_server: Still destroying... [id=i-0026fb0c5d017d88c, 00m40s elapsed]
aws_instance.nginx_server: Destruction complete after 43s
aws_security_group.nginx_sg: Destroying... [id=sg-0890473ca7b7e125d]
aws_security_group.nginx_sg: Destruction complete after 43s
aws_security_group.nginx_sg: Destruction complete after 43s

Destroy complete! Resources: 2 destroyed.

dell@DESKTOP-VVNMU50 MINGW64 /d/Freelancing/Uptwork/Task-2/Task-1-EC2-NGINX
$
```

## **TASK#2:**

## *Task 2: Deploy Backend API on EC2 using Terraform*

*Goal: Automate API deployment with Docker on EC2 via Terraform.*

*Why:* This teaches how to combine Terraform and Docker to automate app deployment.

*What You'll Learn: Writing a Dockerfile, managing Docker Hub images, using Terraform user\_data.*

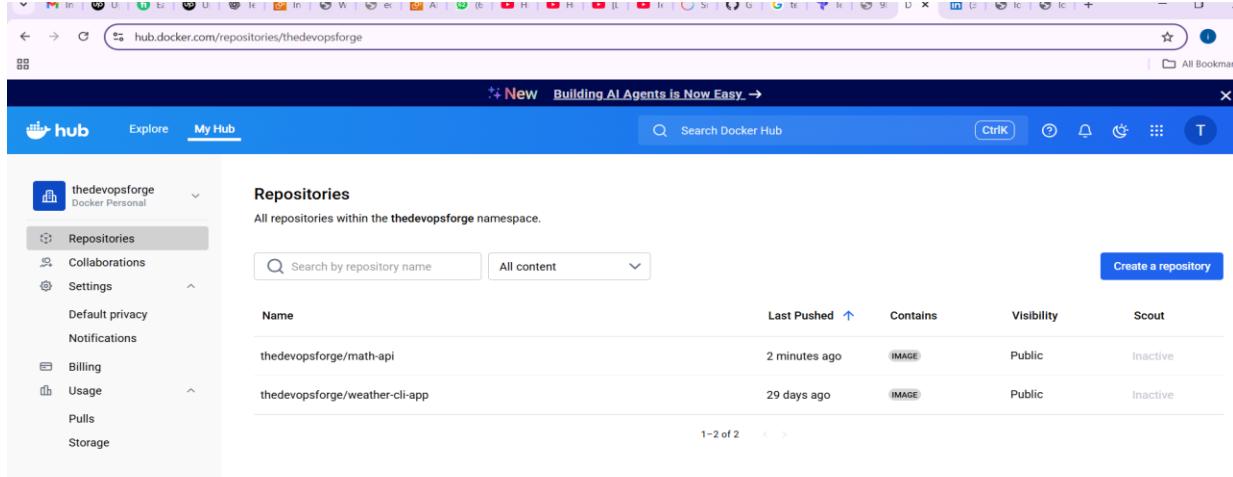
- Write a Dockerfile for a Node.js or Flask API.
  - Push your image to Docker Hub.
  - Use Terraform to create an EC2 instance with Docker installed.
  - Deploy the container using user\_data.
  - Validate via the /health endpoint.

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows files like server.js, package.json, Dockerfile, app, package.json, server.js, Dockerfile, main.tf, and terraform-key.pem.
- Dockerfile:** The current file is Dockerfile, containing the following code:

```
FROM node:18-slim
WORKDIR /app
COPY ./app .
RUN npm install
EXPOSE 80
CMD ["npm", "start"]
```
- Terminal:** The terminal shows the command `docker login` being run, followed by the output "Authenticating with existing credentials... Login Succeeded".
- Output:** The build process is shown in the terminal:

```
[+] Building 1.4s (9/9) FINISHED
--> [internal] load build definition from Dockerfile
--> => transferring dockerfile: 143B
--> [internal] load metadata for docker.io/library/node:18-slim
--> [internal] load .dockerignore
--> => transferring context: 2B
--> [1/4] FROM docker.io/library/node:18-slim@sha256:f9ab18e354e6855ae56ef2bz90dd225c1e51a564f87584b9bd21dd651838830e
--> [internal] load build context
--> => transferring context: 94B
--> CACHED [2/4] WORKDIR /app
--> CACHED [3/4] COPY ./app .
--> CACHED [4/4] RUN npm install
--> exporting to image
--> => exporting layers
--> => writing image sha256:663411f5e64e82feb569e21dd9c6ace82b37f70ef4399ab52b9839eeeeeef68639
--> => naming to docker.io/thedevopsforge/math-api
```
- Right Panel:** Shows the Docker image details for docker:default with a total size of 0.05GB.



```
MINGW64/d/Freelancing/Uwork/Task-2/Task-2-EC2-Math-API
$ ls
app/ Dockerfile main.tf terraform-key.pem

dell@DESKTOP-VV5NU50 MINGW64 /d/Freelancing/Uwork/Task-2/Task-2-EC2-Math-API
$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version for hashicorp/aws...
- Installed hashicorp/aws v6.4.0...
- Installed hashicorp/aws vs 4.0.0 (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,
run this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

dell@DESKTOP-VV5NU50 MINGW64 /d/Freelancing/Uwork/Task-2/Task-2-EC2-Math-API
$ terraform apply

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.math_api_instance will be created
+ resource "aws_instance" "math_api_instance" {
    ami                               = "ami-050fd9796aa387c0d"
    arn                             = (known after apply)
    associate_public_ip_address      = (known after apply)
    availability_zone                = (known after apply)
    disable_api_stop                 = (known after apply)
    disable_api_termination          = (known after apply)
    ebs_optimized                   = (known after apply)
    enable_primary_ipv6              = (known after apply)
    get_password_data               = false
    host_id                          = (known after apply)
    host_resource_group_arn          = (known after apply)
    iam_instance_profile             = (known after apply)
    id                                = (known after apply)
}

Type here to search 33°C Haze 4:24 pm
dell@DESKTOP-VV5NU50 MINGW64 /d/Freelancing/Uwork/Task-2/Task-2-EC2-Math-API
```

```
Select MINGW64/d/Freelancing/Uwork/Task-2/Task-2-EC2-Math-API
  # (1 unchanged attribute hidden)
  },
  + {
    + cidr_blocks           = [
      + "0.0.0.0/0",
    ]
    + from_port            = 80
    + ipv6.cidr_blocks     = []
    + prefix_list_ids     = []
    + protocol             = "tcp"
    + security_groups      = []
    + self                 = false
    + to_port              = 80
    # (1 unchanged attribute hidden)
  },
  + name                  = "math-api-security-group"
  + name_prefix           = (known after apply)
  + owner_id              = (known after apply)
  + region                = "us-east-1"
  + replace_rules_on_delete = false
  + tags_all              = (known after apply)
  + vpc_id                = (known after apply)
}

Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + public_ip = (known after apply)

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

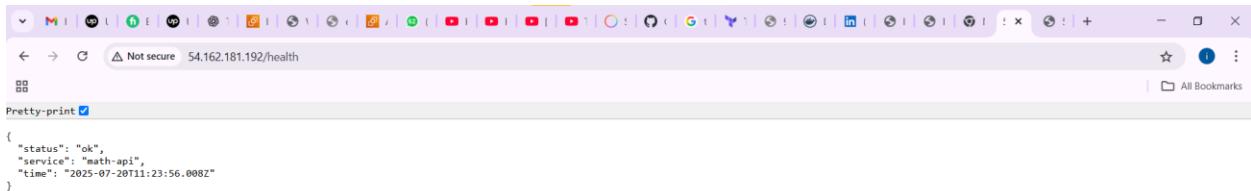
Enter a value: yes

aws_security_group.math_api_sg: Creating...
aws_security_group.math_api_sg: Creation complete after 10s [id=sg-05adaa413b4916144]
aws_instance.math_api_instance: Creating...
aws_instance.math_api_instance: Still creating... [00m10s elapsed]
aws_instance.math_api_instance: Creation complete after 20s [id=i-0ce318a271c2d24c5]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

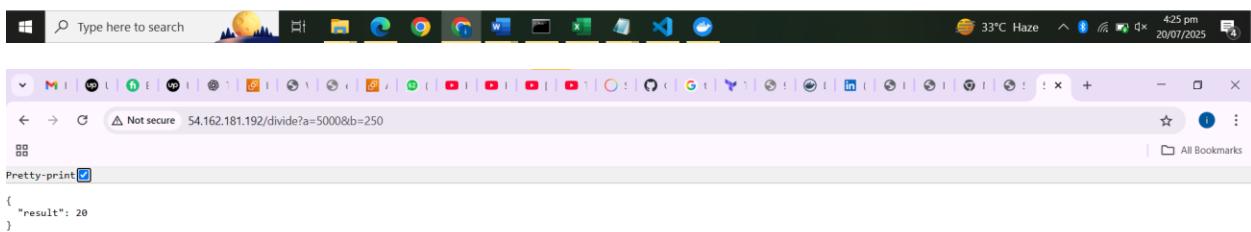
Outputs:
public_ip = "54.162.181.192"

dell@DESKTOP-VV5NU50 MINGW64 /d/Freelancing/Uwork/Task-2/Task-2-EC2-Math-API
$
```



A screenshot of a Windows desktop environment. At the top is a purple taskbar with several pinned icons, including Mail, File Explorer, Microsoft Edge, and others. Below the taskbar is a Microsoft Edge browser window. The address bar shows the URL "54.162.181.192/health". The main content area displays a JSON response with the "Pretty-print" checkbox checked:

```
{  
  "status": "ok",  
  "service": "math-api",  
  "time": "2025-07-20T11:23:56.008Z"  
}
```



A screenshot of a Windows desktop environment. At the top is a purple taskbar with several pinned icons, including Mail, File Explorer, Microsoft Edge, and others. Below the taskbar is a Microsoft Edge browser window. The address bar shows the URL "54.162.181.192/divide?a=5000&b=250". The main content area displays a JSON response with the "Pretty-print" checkbox checked:

```
{  
  "result": 20  
}
```



```
az MINGW64/d/Freelancing/Uwork/Task-2/Task-2-EC2-Math-API
public_ip = "54.162.181.192"
az MINGW64/d/Freelancing/Uwork/Task-2/Task-2-EC2-Math-API
# terraform destroy
aws_security_group.math_api_sg: Refreshing state... [id=sg-05adaa413b4916144]
aws_instance.math_api_instance: Refreshing state... [id=i-0ce318a271c2d24c5]

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.math_api_instance will be destroyed
- resource "aws_instance" "math_api_instance" {
    ami                               = "ami-050fd796aa387c0d" > null
    associate_public_ip_address       = true > null
    availability_zone                = "us-east-1b" > null
    disable_api_stop                 = false > null
    disable_api_termination          = false > null
    ebs_optimized                    = false > null
    get_password_data                = false > null
    hibernation                      = false > null
    id                                = "i-0ce318a271c2d24c5" > null
    instance_initiated_shutdown_behavior = "stop" > null
    instance_state                   = "running" > null
    instance_type                    = "t3.micro" > null
    ip_address_count                = 0 > null
    ipv6_addresses                  = [] > null
    key_name                          = "terraform-key" > null
    monitoring                        = false > null
    placement_partition_number       = 0 > null
    primary_network_interface_id     = "eni-030cbd39fcad93b23" > null
    private_dns                       = "ip-172-31-26-15.ec2.internal" > null
    private_ip                         = "172.31.26.15" > null
    public_dns                         = "ec2-54-162-181-192.compute-1.amazonaws.com" > null
    public_ip                           = "54.162.181.192" > null
    region                            = "us-east-1" > null
    secondary_private_ips             = [] > null
    security_groups {
        "Name" = "math-api-security-group",
    } > null
    source_dest_check                 = true > null
    subnet_id                          = "subnet-026751ad9cc52401f" > null
    tags {
        "Name" = "math-api-instance"
    } > null
}
```

## **TASK#3:**

### *Task 3: Use Terraform Modules*

*Goal: Modularize Terraform code for reusability.*

*Why: Modules help avoid repetition and improve code maintenance.*

*What You'll Learn: How to build and reuse Terraform modules.*

- Create modules for EC2, Security Group, and deployment logic.
  - Use variables and outputs to pass data between modules.
  - Call these modules in your main .tf files.

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

- Explorer View:** Shows the file structure of the Terraform workspace. Opened files include `variables.tf`, `outputs.tf`, `main.tf`, `main.tf` (the active editor), `outputs.tf`, `variables.tf`, and `script.sh`.
- Taskbar:** Displays the current task as "Task 3-Terraform-Modules".
- Editor:** The main code editor displays the `main.tf` configuration file. The code defines an AWS provider, a security group module, user data, and an EC2 instance with specific configurations like AMI, instance type, key pair, and security group.

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

data "aws_vpc" "default" {
  default = true
}

module "sg" {
  source      = "./modules/security_group"
  name        = "modular-sg"
  description = "SG via module"
  vpc_id      = data.aws_vpc.default.id
}

module "user_data" {
  source      = "./modules/user_data"
  docker_image = "thedevopsforge/math-api" # Replace if your image name is different
  container_port = 80
}

module "ec2" {
  source      = "./modules/ec2"
  ami         = "ami-050fd796aa387c0d" # Amazon Linux 2023
  instance_type = "t3.micro"
  key_name    = "terraform-key"
  security_group_id = module.sg.id
  user_data   = module.user_data.user_data
  name        = "modular-ec2"
}
```

```
Administrator: MINGW64 /d/Freelancing/Uwork/Task-2/Task-3-Terraform-Modules
$ terraform init
Initializing the backend...
Initializing modules...
  - ec2 in modules/ec2
  - sg in modules/security_group
  - user_data in modules/user_data
Initializing provider plugins...
  - Finding latest version of hashicorp/aws...
    - Finding latest version of hashicorp/template...
    - Installing hashicorp/aws v6.4.0...
      Installed hashicorp/aws v6.4.0 (signed by HashiCorp)
    - Installing hashicorp/template v2.2.0...
      Installed hashicorp/template v2.2.0 (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,
run this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

Administrator: MINGW64 /d/Freelancing/Uwork/Task-2/Task-3-Terraform-Modules
$ terraform plan
module.user_data.data.template_file.user_data: Reading...
module.user_data.data.template_file.user_data: Read complete after 0s [id=1bb058905b52e46d518fe9d3799cd6a392226f8d38fd35c7c6a00e52f5e03dee]
data.aws_vpc.default: Reading...
data.aws_vpc.default: Read complete after 9s [id=vpc-0699b5de9a5d4840b]

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:

# module.ec2.aws_instance.this will be created
+ resource "aws_instance" "this" {
  + ami                                = "ami-050fd9796aa387c0d"
  + arn                                = "(known after apply)"
  + associate_public_ip_address        = "(known after apply)"
  + availability_zone                   = "(known after apply)"
  + disable_api_stop                    = "(known after apply)"
  + disable_api_termination            = "(known after apply)"
```

```
MINGW64/d/Freelancing/Uptwork/Task-2/Task-3-Terraform-Modules
+ prefix_list_ids = []
+ protocol        = "tcp"
+ security_groups = []
+ self             = false
+ to_port          = 22
# (1 unchanged attribute hidden)
},
{
+ cidr_blocks      = [
+ "0.0.0.0/0",
]
+ from_port         = 80
+ ipv6_cidr_blocks = []
+ prefix_list_ids  = []
+ protocol          = "tcp"
+ security_groups   = []
+ self              = false
+ to_port           = 80
# (1 unchanged attribute hidden)
],
+
+ name               = "modular-sg"
+ name_prefix        = "(known after apply)"
+ owner_id           = "(known after apply)"
+ region             = "us-east-1"
+ revoke_rules_on_delete = false
+ tags_all           = "(known after apply)"
+ vpc_id              = "vpc-0699b5de9a5d4840b"
}

Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ instance_ip = (known after apply)
module.sg.aws_security_group.this: Creating...
module.sg.aws_security_group.this: Creation complete after 9s [id=sg-0a4cdae5dcde2c93e]
module.ec2.aws_instance.this: Creating...
module.ec2.aws_instance.this: Still creating... [0m01s elapsed]
module.ec2.aws_instance.this: Still creating... [0m02s elapsed]
module.ec2.aws_instance.this: Creation complete after 22s [id=i-06c4a933e0f860c8b]

apply complete! Resources: 2 added, 0 changed, 0 destroyed.

outputs:
instance_ip = "13.221.124.85"

Bell@DESKTOP-VVSNNU0 MINGW64 /d/Freelancing/Uptwork/Task-2/Task-3-Terraform-Modules
$
```

```
Not secure 13.221.124.85/subtract?a=3000&b=1000
Pretty-print
{
  "result": 2000
}
```



## Task#4:

*Task 4: Use S3 Backend and State Locking*

*Goal: Manage Terraform state remotely and securely.*

*Why: Remote state sharing is essential in team environments.*

*What You'll Learn: Setting up S3 backend and DynamoDB locking.*

- Create an S3 bucket for storing .tfstate files.
- Create a DynamoDB table for state locking.
- Configure the backend block in Terraform.
- Test state locking with concurrent apply.

The image contains two screenshots of the AWS IAM console interface. Both screenshots show the 'Permissions' tab selected under a user named 'terraform-user'. In the top screenshot, the 'Permissions policies (2)' section is visible, listing two AWS managed policies: 'AmazonEC2FullAccess' and 'AmazonVPCFullAccess', both attached directly. Below this, the 'Permissions boundary (not set)' section is shown. A callout box highlights the 'Add permissions' button. In the bottom screenshot, the user is in the process of creating a new policy. The 'Specify permissions' step is selected, and the 'Policy editor' section shows a JSON editor with a partially defined policy statement. The 'Edit statement' panel on the right is open, showing a 'Select a statement' dropdown and a 'Add new statement' button. The JSON code in the editor is as follows:

```
1  {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Action": [
7                 "s3:createBucket",
8                 "s3:listBucket",
9                 "s3:getBucketLocation",
10                "s3:putObject",
11                "s3:getObject",
12                "s3:deleteObject",
13                "s3:putObjectacl",
14                "s3:listAllMyBuckets"
15            ],
16            "Resource": [
17                "arn:aws:s3:::*",
18                "arn:aws:s3:::/*"
19            ],
20        },
21    ]
```

Screenshot of the AWS IAM 'Create policy' step showing JSON code for a policy named 'TerraformS3DynamonlinePolicy'. The policy grants permissions for S3 and DynamoDB actions on specific resources.

```

10      "s3:PutObject",
11      "s3:GetObject",
12      "s3:DeleteObject",
13      "s3:PutObjectAcl",
14      "s3:ListAllMyBuckets"
15    ],
16    "Resource": [
17      "arn:aws:s3:::*",
18      "arn:aws:s3:::/*"
19    ],
20  },
21  {
22    "Effect": "Allow",
23    "Action": [
24      "dynamodb:CreateTable",
25      "dynamodb:DescribeTable",
26      "dynamodb:PutItem",
27      "dynamodb:GetItem",
28      "dynamodb:DeleteItem",
29      "dynamodb:ListTables"
30    ]
31  }
32]

```

**Add new statement**

JSON Ln 38, Col 0      1556 of 2048 characters remaining

Security: 0 Errors: 0 Warnings: 0 Suggestions: 1

Cancel    Next

Screenshot of the AWS IAM 'Create policy' step showing the 'Review and create' section. The policy name is 'TerraformS3DynamonlinePolicy'. The 'Permissions defined in this policy' table shows two entries: 'DynamoDB' with 'Limited: List, Read, Write' access level and 'All resources' resource, and 'S3' with 'Limited: List, Permissions management, Read, Write' access level and 'Multiple' resource.

Service	Access level	Resource	Request condition
DynamoDB	Limited: List, Read, Write	All resources	None
S3	Limited: List, Permissions management, Read, Write	Multiple	None

**Create policy**

Screenshot of the AWS IAM User Details page for 'terraform-user' in the 'us-east-1' region. The user was created on July 20, 2025, at 12:49 (UTC+00:00). The 'Identity and Access Management (IAM)' section shows the ARN: arn:aws:iam::400375465965:user/terraform-user. The 'Access management' section lists 'User groups', 'Roles', 'Policies', 'Identity providers', and 'Account settings'. The 'Access reports' section includes 'Access Analyzer', 'Unused access', 'Analyzer settings', 'Credential report', 'Organization activity', and 'Service control policies'. The 'Permissions' tab is selected, showing three attached policies: 'AmazonEC2FullAccess', 'AmazonVPCFullAccess', and 'TerraformS3DynamoInlinePolicy'. The 'Security credentials' tab shows two access keys: 'Access key 1' (AKIAV2OB4Y7W7OY6D6M3 - Active, used 18 days ago, 19 days old) and 'Access key 2' (Create access key). The 'Last Accessed' tab is also present.

```
C:\Users\Del\aws s3 mb s3://insia-terraform-state-bucket-2025 --region us-east-1
make_bucket: insia-terraform-state-bucket-2025

C:\Users\Del\aws s3api put-bucket-versioning --bucket insia-terraform-state-bucket-2025 --versioning-configuration Status=Enabled --region us-east-1

C:\Users\Del\aws s3api put-bucket-encryption --bucket insia-terraform-state-bucket-2025 --server-side-encryption-configuration "{\"Rules\":[(\"ApplyServerSideEncryptionByDefault\":{\"SSEAlgorithm\":\"AES256\"})]} --region us-east-1

C:\Users\Del\aws dynamodb create-table ^
--table-name terraform-locks ^
--attribute-definitions AttributeName=LockID,AttributeType=S ^
--key-schema AttributeName=LockID,KeyType=HASH ^
--billing-mode PAY_PER_REQUEST ^
--region us-east-1

{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "LockID",
                "AttributeType": "S"
            }
        ],
        "TableName": "terraform-locks",
        "KeySchema": [
            {
                "AttributeName": "LockID",
                "KeyType": "HASH"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2025-08-09T00:01:37.178000+05:00",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 0,
            "WriteCapacityUnits": 0
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-east-1:400375465965:table/terraform-locks",
        "TableId": "e481c49f-10ab-4a8e-b50d-3c0666adeff",
        "BillingModeSummary": {
            "BillingMode": "PAY_PER_REQUEST"
        },
        "DeletionProtectionEnabled": false
    }
}
```

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

- File Explorer:** Shows a project structure under "OPEN EDITORS" with files like "main.tf", ".terraform.lock.hcl", and "main.tf.lock".
- Terminal:** Taskbar tab labeled "Task-2". Command history shows:
 

```
PS D:\Freelancing\Upwork\Task-2\Task4-Remote-State> terraform init
Terraform has been successfully initialized!
```
- Output:** Shows the Terraform initialization log.
- Status Bar:** Displays file path "D:\Freelancing\Upwork\Task-2\Task4-Remote-State> main.tf", line 33, column 1, and system status like "32°C Haze".

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

- File Explorer:** Shows a project structure under "OPEN EDITORS" with files like "main.tf", ".terraform.lock.hcl", and "main.tf.lock".
- Terminal:** Taskbar tab labeled "Task-2". Command history shows:
 

```
PS D:\Freelancing\Upwork\Task-2> cd Task4-Remote-State
PS D:\Freelancing\Upwork\Task-2\Task4-Remote-State> terraform apply
Acquiring state lock. This may take a few moments...
```
- Output:** Shows the Terraform apply command and an error message about acquiring the state lock.
- Details:** A red box highlights the error message:
 

```
Error: Error acquiring the state lock
```

Error message: operation error DynamoDB: PutItem, https response error StatusCode: 400, RequestID: MNULCV00P5JTA2I1TEP1DCB0UWV4KQNS0AEMVJF66Q9ASUAJ6, ConditionalCheckFailedException: The conditional request failed
- Status Bar:** Displays file path "D:\Freelancing\Upwork\Task-2\Task4-Remote-State> main.tf", line 9, column 2, and system status like "32°C Haze".

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

- File Explorer:** Shows files like `main.tf`, `Task-1-EC2-NGINX`, `Task-2-E2-Math-API`, `Task-3-Terrafor...`, `Task4-Remote-S...`, `terraform`, and `terraform.locks`.
- Code Editor:** Displays `main.tf` with Terraform configuration for AWS S3 and DynamoDB.
- Terminal:** Shows the command `terraform destroy -auto-approve` being run, followed by the execution plan and confirmation of destruction.
- Status Bar:** Includes file navigation, workspace status, and system information (32°C Haze, 1227 am, 09/08/2025).

The screenshot shows a Windows Command Prompt window with the following content:

```
C:\Users\DELL>aws s3 rb s3://insia-terraform-state-bucket-2025 --force
delete: s3://insia-terraform-state-bucket-2025/terraform.tfstate
remove_bucket failed: s3://insia-terraform-state-bucket-2025 An error occurred (AccessDenied) when calling the DeleteBucket operation: User: arn:aws:iam::400375465965:user/terraform-user is not authorized to perform: s3:DeleteBucket on resource: "arn:aws:s3:::insia-terraform-state-bucket-2025" because no identity-based policy allows the s3:DeleteBucket action
C:\Users\DELL>
C:\Users\DELL>aws dynamodb delete-table --table-name terraform-locks --region us-east-1
{
    "TableDescription": {
        "TableName": "terraform-locks",
        "TableStatus": "DELETING",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 0,
            "WriteCapacityUnits": 0
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-east-1:400375465965:table/terraform-locks",
        "TableId": "e481c49f-10ab-4a8e-b5d0-3c8606adef1f",
        "BillingModeSummary": {
            "BillingMode": "PAY_PER_REQUEST",
            "LastUpdateToPayPerRequestDateTime": "2025-08-09T00:01:37.178000+05:00"
        },
        "DeletionProtectionEnabled": false
    }
}
C:\Users\DELL>
```

The status bar at the bottom shows system information (32°C Haze, 1226 am, 09/08/2025).

## Task#5:

## *Task 5: Multi-Environment Support*

*Goal: Manage multiple deployment environments.*

*Why: Dev, staging, and prod environments require separation.*

## *What You'll Learn: Using Terraform workspaces and conditional logic.*

- Create workspaces for dev, staging, and prod.
  - Use conditional variables to modify infrastructure per workspace.
  - Deploy different instance types per environment.

The screenshot shows a Windows desktop environment with a code editor and a terminal window.

**Code Editor (VS Code):**

- Left Sidebar:** Explorer, Open Editors, Task List, Task View.
- Open Editors:** main.tf, outputs.tf, variables.tf, provider.tf.
- Terminal (Task View):**
  - Command Prompt output:

```
C:\Users\...>aws dynamodb create-table --table-name terraform-locks --attribute-definitions AttributeName=LockID,AttributeType=S --key-schema AttributeName=LockID,KeyType=HASH --provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1 --region us-east-1
{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "LockID",
        "AttributeType": "S"
      }
    ],
    "TableName": "terraform-locks",
    "KeySchema": [
      {
        "AttributeName": "LockID",
        "KeyType": "HASH"
      }
    ],
    "TableStatus": "CREATING",
    "CreationDateTime": "2025-08-09T14:36:02.840000+05:00",
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 1,
      "WriteCapacityUnits": 1
    },
    "TableSizeBytes": 0,
    "ItemCount": 0,
    "TableArn": "arn:aws:dynamodb:us-east-1:400375405965:table/terraform-locks",
    "TableId": "1b6d08990-ae74-4775-980a-dafc8027cd0d",
    "DeletionProtectionEnabled": false
  }
}
```
  - aws s3api create-bucket --bucket insia-terraform-state-bucket-2025 --region us-east-1
 {
 "Location": "/insia-terraform-state-bucket-2025"
 }
  - aws s3api put-bucket-versioning --bucket insia-terraform-state-bucket-2025 --versioning-configuration Status=Enabled --region us-east-1
  - aws s3api put-bucket-encryption --bucket insia-terraform-state-bucket-2025 --server-side-encryption-configuration '{"Rules":[{"ApplyServerSideEncryptionByDefault":{"SSEAlgorithm":"AES256"}]}]' --region us-east-1
  - Error parsing parameter '--server-side-encryption-configuration': Expected '=', received: ''' for input:
 'Rules:[(ApplyServerSideEncryptionByDefault:(SSEAlgorithm:AES256))]'
  - aws s3api put-bucket-encryption --bucket insia-terraform-state-bucket-2025 --server-side-encryption-configuration "{\"Rules\":[(\"ApplyServerSideEncryptionByDefault\":(\"SSEAlgorithm\":\"AES256\"))]} --region us-east-1

File Edit Selection View Go Run Terminal Help Task-2

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE

**OPEN EDITORS**

- main.tf
- outputs.tf
- variables.tf
- provider.tf

**TASK-2**

- Task-1-EC2-NGINX
- Task-2-EC2-Math-API
- Task-3-Terrafor...
- Task4-Remote-S...
- Task5-MultiEnv
- terrafrom
- terraform.lock.hcl
- main.tf
- outputs.tf
- provider.tf
- variables.tf

PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> **terraform init**

Initializing the backend...

Successfully configured the backend "s3"! Terraform will automatically use this backend unless the backend configuration changes.

Initializing provider plugins...

- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.8.0...

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.

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.

rerun this command to reinitialize your working directory. If you forget, other PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> **terraform workspace new dev**

**Created and switched to workspace "dev"**

You're now on a new, empty workspace. Workspaces isolate their state, so if you run "terraform plan" Terraform will not see any existing state

PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> **terraform workspace new staging**

You're now on a new, empty workspace. Workspaces isolate their state, so if you run "terraform plan" Terraform will not see any existing state for this configuration.

PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> **terraform workspace show prod**

PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> **terraform workspace list**

default

dev

\* prod

staging

Ln 21, Col 1 Spaces: 4 UTF-8 CRLF Plain Text Finish Setup Go Live 248 pm 28°C Smoke 09/08/2025

File Edit Selection View Go Run Terminal Help Task-2

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE

**OPEN EDITORS**

- main.tf
- outputs.tf
- variables.tf
- provider.tf

**TASK-2**

- Task-1-EC2-NGINX
- Task-2-EC2-Math-API
- Task-3-Terrafor...
- Task4-Remote-S...
- Task5-MultiEnv
- terrafrom
- terraform.lock.hcl
- main.tf
- outputs.tf
- provider.tf
- variables.tf

PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> **terraform workspace select dev**

**Switched to workspace "dev".**

PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> **terraform apply --auto-approve**

Acquiring state lock. This may take a few moments...

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.example will be created
+ resource "aws_instance" "example" {
    + ami = "ami-0c55b159cbfafef0"
    + arm = (known after apply)
    + associate_public_ip_address = (known after apply)
    + availability_zone = (known after apply)
    + disable_api_stop = (known after apply)
    + disable_api_termination = (known after apply)
    + ebs_optimized = (known after apply)
    + enable_primary_ipv6 = (known after apply)
    + force_destroy = false
    + get_password_data = false
    + host_id = (known after apply)
    + host_resource_group_arn = (known after apply)
    + id = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
    + instance_lifecycle = (known after apply)
    + instance_state = (known after apply)
    + instance_type = "t2.micro"
    + ipv6_address_count = (known after apply)
    + ipv6_addresses = (known after apply)
    + key_name = (known after apply)
    + monitoring = (known after apply)
    + outpost_arn = (known after apply)
    + password_data = (known after apply)
    + placement_group = (known after apply)
    + placement_partition_number = (known after apply)
    + primary_network_interface_id = (known after apply)
    + private_dns = (known after apply)
    + private_ip = (known after apply)
}
```

Ln 21, Col 1 Spaces: 4 UTF-8 CRLF Plain Text Finish Setup Go Live 249 pm 28°C Smoke 09/08/2025

```
File Edit Selection View Go Run Terminal Help Task-2 OPEN EDITORS main.tf Task-2 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE powerhell + ... x Switched to workspace "staging". PS D:\Freelancing\Upwork\Task-2\Tasks-MultiEnv terraform apply -auto-approve Acquiring state lock. This may take a few moments... 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.example will be created + resource "aws_instance" "example" { = "ami-0c55b159cbfafef0" + ami = (known after apply) + arm = (known after apply) + associate_public_ip_address = (known after apply) + availability_zone = (known after apply) + disable_api_stop = (known after apply) + ebs_optimized = (known after apply) + enable_primary_ipv6 = (known after apply) + force_destroy = false + get_password_data = false + host_id = (known after apply) + host_resource_group_arn = (known after apply) + iam_instance_profile = (known after apply) + id = (known after apply) + instance_initiated_shutdown_behavior = (known after apply) + instance.lifecycle = (known after apply) + instance_state = (known after apply) + instance_type = "t2.small" + ipv6_address_count = (known after apply) + ipv6_addresses = (known after apply) + key_name = (known after apply) + monitoring = (known after apply) + outpost_arn = (known after apply) + password_data = (known after apply) + placement_group = (known after apply) + placement_partition_number = (known after apply) + primary_network_interface_id = (known after apply) + private_dns = (known after apply) + private_ip = (known after apply) + public_dns = (known after apply) 
```

```
File Edit Selection View Go Run Terminal Help Task-2 OPEN EDITORS main.tf Task-2 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE powerhell + ... x Switched to workspace "prod". PS D:\Freelancing\Upwork\Task-2\Tasks-MultiEnv terraform workspace select prod Acquiring state lock. This may take a few moments... 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.example will be created + resource "aws_instance" "example" { = "ami-0c55b159cbfafef0" + ami = (known after apply) + arm = (known after apply) + associate_public_ip_address = (known after apply) + availability_zone = (known after apply) + disable_api_stop = (known after apply) + ebs_optimized = (known after apply) + enable_primary_ipv6 = (known after apply) + force_destroy = false + get_password_data = false + host_id = (known after apply) + host_resource_group_arn = (known after apply) + iam_instance_profile = (known after apply) + id = (known after apply) + instance_initiated_shutdown_behavior = (known after apply) + instance.lifecycle = (known after apply) + instance_state = (known after apply) + instance_type = "t2.medium" + ipv6_address_count = (known after apply) + ipv6_addresses = (known after apply) + key_name = (known after apply) + monitoring = (known after apply) + outpost_arn = (known after apply) + password_data = (known after apply) + placement_group = (known after apply) + placement_partition_number = (known after apply) + primary_network_interface_id = (known after apply) + private_dns = (known after apply) + private_ip = (known after apply) 
```

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with navigation links like 'Amazon S3', 'General purpose buckets', 'Storage Lens', and 'Feature spotlight'. The main area displays the contents of the 'env:' bucket. It shows three objects: 'dev/' (Folder), 'prod/' (Folder), and 'staging/' (Folder). There are buttons for 'Copy S3 URI', 'Copy URL', 'Download', 'Open', 'Delete', 'Actions', 'Create folder', and 'Upload'.

The screenshot shows the VS Code interface with multiple terminal tabs open. The tabs are labeled 'Task-2', 'Task-3', 'Task-4', and 'Task5-MultiEnv'. Each tab shows the output of Terraform commands related to 'destroy' operations. For example, the 'Task-2' tab shows:

```

PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> terraform destroy -auto-approve
Acquiring state lock. This may take a few moments...
Changes to Outputs:
You can apply this plan to save these new output values to the Terraform state, without changing any real infrastructure.
Releasing state lock. This may take a few moments...
Destroy complete! Resources: 0 destroyed.
PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv>
PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> terraform workspace select staging
Switched to workspace "staging".
PS D:\Freelancing\Upwork\Task-2\Task5-MultiEnv> terraform destroy -auto-approve
Acquiring state lock. This may take a few moments...

```

The other tabs show similar outputs for their respective tasks.

Destroy the S3 and DynamoDB as well.

## Task#6:

*Task 6: CI/CD with GitHub Actions for Terraform*

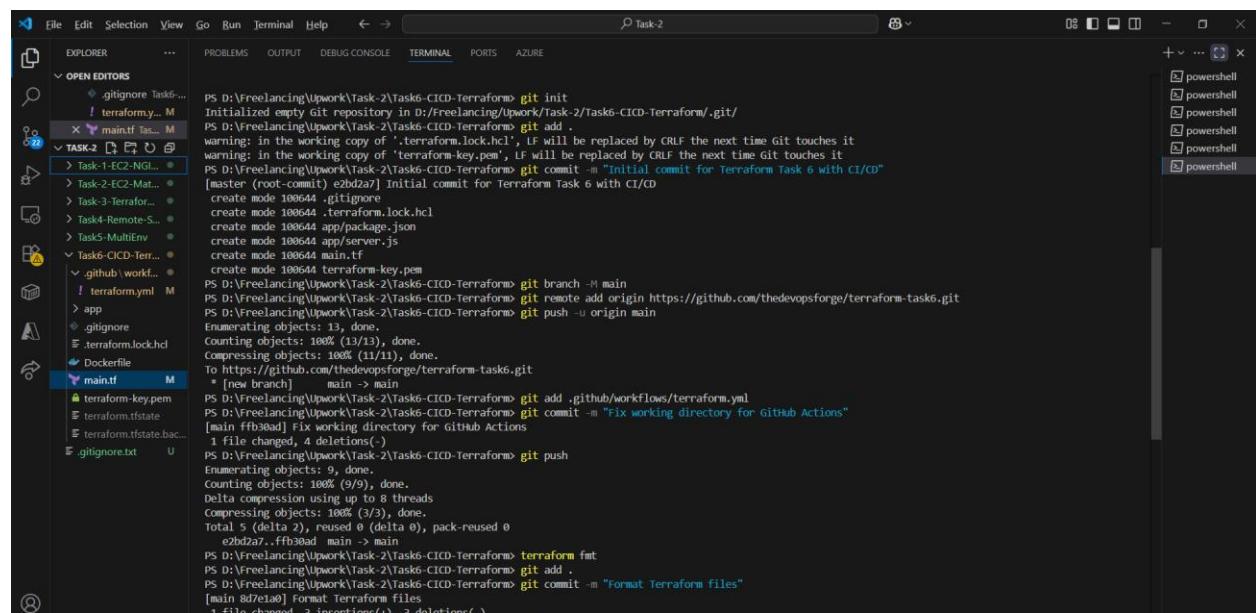
*Goal: Automate your Terraform workflow with CI/CD.*

*Why: Automation ensures consistency and avoids manual errors.*

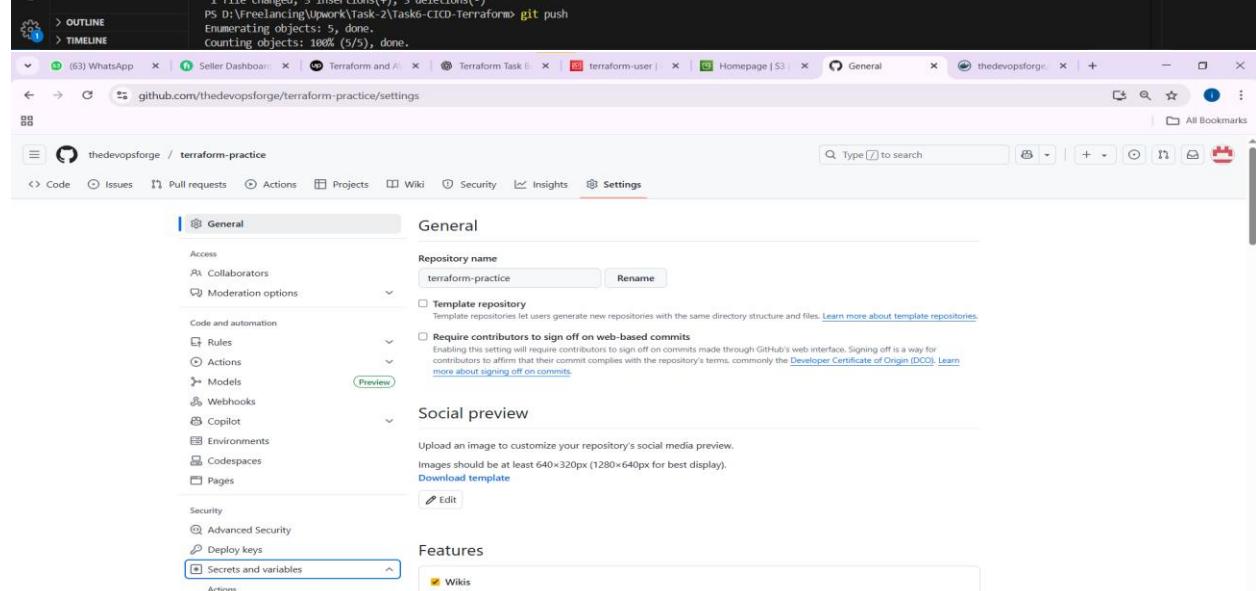
*What You'll Learn: Creating CI/CD pipelines, managing secrets, running plans and applies automatically.*

- Write a GitHub Actions workflow to plan and apply Terraform scripts.
- Use GitHub Secrets to store credentials securely.
- Run Terraform plan on PRs, apply on main branch merges.

Using task 2s code in task6.



```
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git init
Initialized empty Git repository in D:/Freelancing/Upwork/Task-2/Task6-CI_CD-Terraform/.git/
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git add .
warning: in the working copy of '.terraform.lock.hcl', LF will be replaced by CRLF the next time git touches it
warning: in the working copy of 'terraform-key.pem', LF will be replaced by CRLF the next time git touches it
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git commit -m "Initial commit for Terraform Task 6 with CI/CD"
[master (root-commit) e2bd2a7] Initial commit for Terraform Task 6 with CI/CD
  create mode 100644 .gitignore
  create mode 100644 .terraform.lock.hcl
  create mode 100644 app/package.json
  create mode 100644 app/server.js
  create mode 100644 main.tf
  create mode 100644 terraform-key.pem
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git branch -M main
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git remote add origin https://github.com/theoppsforge/terraform-task6.git
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git push -u origin main
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Compressing objects: 100% (11/11), done.
To https://github.com/theoppsforge/terraform-task6.git
 * [new branch]      main > main
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git add .github/workflows/terraform.yml
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git commit -m "Fix working directory for GitHub Actions"
[main ffb3ead] Fix working directory for GitHub Actions
 1 file changed, 4 deletions(-)
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
 02bd2a7, ff3b3ad main > main
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> terraform fmt
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git add .
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git commit -m "Format Terraform files"
[main 8d7e1a0] Format Terraform files
 1 file changed, 3 insertions(+), 3 deletions(-)
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.


```

The screenshot shows the GitHub Actions secrets and variables settings page for a repository. The left sidebar lists various settings categories like General, Access, Collaborators, Moderation options, Code and automation, Security, and more. Under the 'Actions' category, the 'Secrets and variables' option is selected. The main content area is titled 'Actions secrets and variables'. It contains sections for 'Environment secrets' and 'Repository secrets'. Both sections indicate that they have no secrets and provide a 'Manage environment secrets' or 'New repository secret' button.

This screenshot shows the same GitHub Actions secrets and variables settings page, but with populated data. In the 'Repository secrets' section, there are three secrets listed: 'AWS\_ACCESS\_KEY\_ID', 'AWS\_REGION', and 'AWS\_SECRET\_ACCESS\_KEY'. Each secret has a name, last updated time ('1 minute ago' or 'now'), and edit/ delete icons.

Name	Last updated
AWS_ACCESS_KEY_ID	1 minute ago
AWS_REGION	now
AWS_SECRET_ACCESS_KEY	now

github.com/theoppsforge/terraform-task6/actions

theoppsforge / terraform-task6

All workflows

Actions

New workflow

Event Status Branch Actor

Format Terraform files

Terraform CI/CD #3: Commit 8d7e1a0 pushed by theoppsforge

Fix working directory for GitHub Actions

Terraform CI/CD #2: Commit ffb30ad pushed by theoppsforge

Initial commit for Terraform Task 6 with CI/CD

Terraform CI/CD #1: Commit e2bd5a7 pushed by theoppsforge



github.com/theoppsforge/terraform-task6/actions/runs/16861057549/job/47761403582

theoppsforge / terraform-task6

Actions

Re-run all jobs Latest #3

Summary

Jobs

terraform

Run details

Usage

Workflow file

terraform

succeeded 4 minutes ago in 34s

Search logs

Set up job

Checkout code

Setup Terraform

Terraform Init

Terraform Format

Terraform Validate

Terraform Plan

Terraform Apply

Post Checkout code

Complete job



github.com/theopendevopsforge/terraform-task6/actions/runs/16861057549/job/47761403582

**terraform**  
succeeded 2 minutes ago in 34s

**Jobs**

- terraform**
- Run details
- Usage
- Workflow file

```

Terraform Apply
156 Changes to Outputs:
157 + public_ip = (known after apply)
158 aws_security_group.math_api_sg: Creating...
159 aws_security_group.math_api_sg: Creation complete after 3s [id=sg-0a0c726f512a6304d]
160 aws_instance.math_api_instance: Creating...
161 aws_instance.math_api_instance: Still creating... [10s elapsed]
162 aws_instance.math_api_instance: Creation complete after 12s [id=i-0d4b23c09c78f780b]
163 aws_instance.math_api_instance: Creation complete after 12s [id=i-0d4b23c09c78f780b]
164
165 Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
166
167 Outputs:
168 public_ip = "3.91.73.12"

```

**Post Checkout code**

```

1 Post job cleanup.
2 /usr/bin/git version
3 git version 2.50.1
4 Temporarily overriding HOME='/home/runnner/work/_temp/a59c936d-de83-4411-b462-dc13df2bafe9' before making global git config changes
5 Adding repository directory to the temporary git global config as a safe directory
6 /usr/bin/git config --global --add safe.directory /home/runnner/work/terraform-task6/terraform-task6
7 /usr/bin/git config --global --unset-all 'core.sshCommand'
8 /usr/bin/git submodule foreach --local --name-only --get-regexp core.sshCommand & git config --local --unset-all 'core.sshCommand' || :
9 /usr/bin/git config --local --name-only --get-regexp http://https://github.com/.extraheader
10 http://https://github.com/.extraheader
11 /usr/bin/git config --local --unset-all http://https://github.com/.extraheader
12 /usr/bin/git submodule foreach --recursive sh -c "git config --local --name-only --get-regexp 'http|https://github.com/.extraheader' & git config --local --unset-all 'http|https://github.com/.extraheader' || :"

```

**Complete job**

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:

**EC2** Instances

**Instances**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public
math-api-inst...	i-0d4b23c09c78f780b	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1c	ec2-3-91-73-12.compute...	3.91.72

Select an instance

CloudShell Feedback

```
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
ffdb3a7..ffbb3ad main --> main
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> terraform fmt
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git add .
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git commit -m "Format Terraform files"
[main 8d7e1a0] Format Terraform files
 1 file changed, 3 insertions(+), 3 deletions(-)
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git push
[main 8d7e1a0] Format Terraform files
 1 file changed, 3 insertions(+), 3 deletions(-)
PS D:\Freelancing\Upwork\Task-2\Task6-CI_CD-Terraform> git push origin update-instance-type
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 312 bytes | 312.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
remote:
remote: Create a pull request for 'update-instance-type' on GitHub by visiting:
remote:   https://github.com/theoppsforge/terraform-task6/pull/new/update-instance-type
remote:
remote: To https://github.com/theoppsforge/terraform-task6.git
 * [new branch]      update-instance-type -> update-instance-type
Ln 33, Col 30 Spaces: 2 UTF-8 CRLF Plain Text Finish Setup Go Live
```

github.com/theoppsforge/terraform-task6

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

terraform-task6 Public

Compare & pull request

main 2 Branches 0 Tags

K200265-Insa-Farhan Format Terraform files 8d7e1a0 - 15 minutes ago 3 Commits

.github/workflows Fix working directory for GitHub Actions 17 minutes ago

app Initial commit for Terraform Task 6 with CI/CD 20 minutes ago

.gitignore Initial commit for Terraform Task 6 with CI/CD 20 minutes ago

.terraform.lock.hcl Initial commit for Terraform Task 6 with CI/CD 20 minutes ago

Dockerfile Initial commit for Terraform Task 6 with CI/CD 20 minutes ago

main.tf Format Terraform files 15 minutes ago

terraform-key.pem Initial commit for Terraform Task 6 with CI/CD 20 minutes ago

README

About No description, website, or topics provided.

Activity 0 stars 0 watching 0 forks

Releases No releases published Create a new release

Packages No packages published Publish your first package

Languages JavaScript 55.8% HCL 40.8% Dockerfile 3.4%

The screenshot shows a web browser window with multiple tabs open. The active tab is 'github.com/thedevopsforge/terraform-task6/actions'. The page displays a list of 'All workflows' under the 'Actions' tab. There are four workflow runs listed:

Workflow Run	Event	Status	Branch	Actor
Change instance type to t2.small #4	update-instance-type	1 minute ago	t2.small	...
Format Terraform files #3	main	17 minutes ago	main	...
Fix working directory for GitHub Actions #2	main	18 minutes ago	main	...
Initial commit for Terraform Task 6 with CI/CD #1	main	21 minutes ago	main	...

The screenshot shows a web browser window with multiple tabs open. The active tab is 'github.com/thedevopsforge/terraform-task6/actions/runs/16861197068/job/47761622258'. The page displays the details for a specific job: 'Change instance type to t2.small #4'. The job summary shows it succeeded 1 minute ago in 18s. The main content area shows the steps of the 'terraform' workflow:

Step	Duration
Set up job	1s
Checkout code	0s
Setup Terraform	1s
Terraform Init	8s
Terraform Format	0s
Terraform Validate	2s
<b>Terraform Plan</b>	3s
Terraform Apply	0s
Post Checkout code	0s
Complete job	0s