

# Basic Infrastructure Management with Terraform

## Introduction:-

### Case Study Overview

This case study focuses on the automation of infrastructure provisioning using Terraform on Amazon Web Services (AWS). Specifically, it involves creating an EC2 (Elastic Compute Cloud) instance and an S3 (Simple Storage Service) bucket, while also demonstrating how to store the EC2 instance's IP address in the S3 bucket. This approach not only streamlines resource management but also enhances reproducibility and version control of the infrastructure.

### Key Feature and Application

The key feature of this case study is the use of Terraform's Infrastructure as Code (IaC) capabilities. Terraform allows users to define infrastructure in a declarative manner, enabling version control, easier collaboration, and automated deployments. This practical use case exemplifies how teams can manage cloud resources efficiently and effectively without the need for manual intervention, minimizing human error.

## Step-by-Step Explanation:-

### Step 1: Initial Setup

#### 1. Prerequisites:

- Ensure that Terraform is installed on your local machine. You can download it from the official Terraform website.
- Set up an AWS account if you don't have one already.
- Configure your AWS credentials using the AWS CLI.

**Create a Project Directory:** Open your terminal and create a directory for your Terraform project:

```
mkdir terraform-aws-ec2-s3
```

```
cd terraform-aws-ec2-s3
```

Or you can also make 2 separate folders for EC2 and S3

### Step 2: Write Terraform Configuration

1. **Create a `main.tf` File in aws-ec2 folder:** Create a file named `main.tf` in your project directory and add the following Terraform configuration:

**Code:-**

```
resource "aws_instance" "myserver" {
  ami = "ami-0866a3c8686eaeeba"
  instance_type = "t2.micro"
  tags = {
    Name = "sample server ad-dev"
  }
}
```

Same way create an `output.tf` and `variables.tf` in the **aws-ec2 folder**

**Code:-** #`output.tf`

```
output "ec2_public_ip" {
  value = aws_instance.myserver.public_ip
}
```

```
output "ec2_instance_type" {
  value = aws_instance.myserver.instance_type
}
```

#`variables.tf`

```
variable "instance_type" {
  description = "value"
  type = string
  default = "t2.micro"
}
```

2. **Create a `main.tf` File in aws-s3 folder:** Create a file named `main.tf` in your project directory and add the following Terraform configuration:

**Code:-**

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.64.0"
    }
  }
}
```

```
resource "aws_s3_bucket"
"demo-bucket-advdevops-practical-2024-sawant-53-d15a" {
  bucket = "demo-bucket-advdevops-practical-2024-sawant-53-d15a"
}
```

```
resource "aws_s3_object" "bucket-data" {  
  bucket =  
aws_s3_bucket.demo-bucket-advdevops-practical-2024-sawant-53-d15a.bucket  
  source = "./myfile.txt"  
  key = "newfile.txt"  
}
```

### Step 3: Initialize Terraform

Run the following command to initialize Terraform, which downloads the necessary provider plugins:

```
terraform init
```

### Step 4: Apply the Configuration

Apply the configuration to create the resources defined in `main.tf`:

```
terraform apply
```

You will see a prompt to confirm the changes; type `yes` to proceed. Terraform will provision the EC2 instance and S3 bucket.

### Step 5: Verify the Resources

After the provisioning completes, you can verify that the EC2 instance and S3 bucket have been created successfully in the AWS Management Console.

### Step 6: Output the EC2 IP Address

The IP address of the EC2 instance will be displayed in the terminal as per the output defined in `main.tf`. You can also check the contents of the S3 bucket to see the `newfile.txt` file containing the IP address.

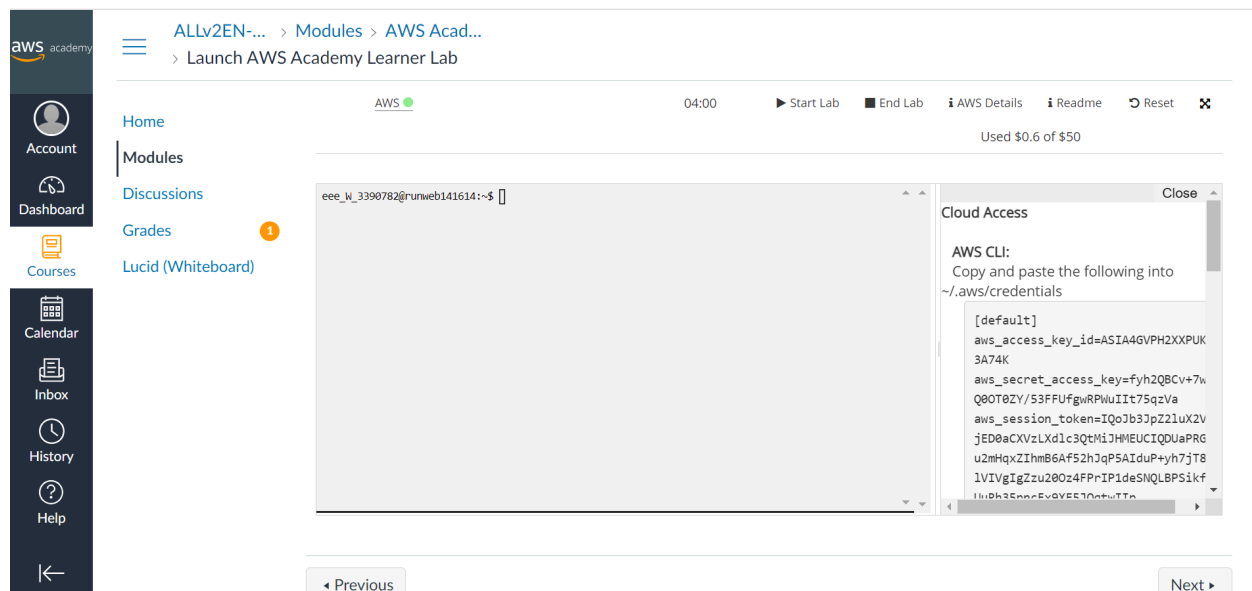
### Guidelines:-

- **Best Practices:**
  - Use version control (e.g., Git) for managing your Terraform code.
  - Regularly check your AWS resource usage to avoid unexpected costs.
  - Tag your resources for better organization and management.
- **Common Errors:**
  - Ensure that the S3 bucket name is unique globally.

- Validate your Terraform configuration using `terraform validate` before applying or you figure out the error in “terraform plan”
- Make sure to check official documentation so that it would be easy to write refer it and write the code.

## Screenshots:-

1. Turn on the aws-academy lab and click on AWS Details to get the access key, secret key, session token



2. Copy all those details in notepad, and make sure to do it like this export  
←Name→="←actual key→".

Name:- Arnav .S. Sawant

D15A

Roll-no:- 53

```
*Untitled - Notepad
File Edit Format View Help
aws_access_key_id=ASIA4GVPH2XXMHYWCBF
aws_secret_access_key=5CKhhN/Y1JwHduoBCB+ELP5wB4RI4175n/CP8+Ip
aws_session_token=IQoJb3JpZ2luX2VjED0aCXVzLXdlc3QtMiJIMEYCIQDNBft/8oUqAA/TYV+CR10ePbqZmfNHeIgUstf5X5ghZAIhAL6u760y9

export AWS_ACCESS_KEY_ID="ASIA4GVPH2XXMHYWCBF"
export AWS_SECRET_ACCESS_KEY="5CKhhN/Y1JwHduoBCB+ELP5wB4RI4175n/CP8+Ip"
export AWS_SESSION_TOKEN="IQoJb3JpZ2luX2VjED0aCXVzLXdlc3QtMiJIMEYCIQDNBft/8oUqAA/TYV+CR10ePbqZmfNHeIgUstf5X5ghZAIhAL6u760y9"
```

### 3. Execute those commands on the CLI.

The screenshot shows the AWS Academy Lab interface. On the left is a sidebar with navigation links: Account, Dashboard, Courses, Calendar, Inbox, History, and Help. The main area displays the lab title 'ALLv2EN-...' and 'Modules > AWS Acad...'. Below this, there's a 'Launch AWS Academy Learner Lab' button. The terminal window shows the following commands and output:

```
eee_w_3390782@runweb141609:~$ export AWS_ACCESS_KEY_ID="ASIA4GVPH2XXMHYWCBF"
eee_w_3390782@runweb141609:~$ export AWS_SECRET_ACCESS_KEY="5CKhhN/Y1JwHduoBCB+ELP5wB4RI4175n/CP8+Ip"
eee_w_3390782@runweb141609:~$ export AWS_SESSION_TOKEN="IQoJb3JpZ2luX2VjED0aCXVzLXdlc3QtMiJIMEYCIQDNBft/8oUqAA/TYV+CR10ePbqZmfNHeIgUstf5X5ghZAIhAL6u760y9"
Ckx/////////wEQARoM0M4OTUS0Dc5NijyYjgzCkTYtsIQe570Hzeoqiwi9M2gnZ70AXLVQXqB+f4ZB8/06vP1w98wZ1A/1+mfJgSAhu1wLc5g03XwftuQe2B4anG1nvqjJrgf4M/6PKZ9Haxox2pn1HCLQErgxf0k1Z/6f8vxs/B6A0Z/RM1Xm1NLUy0dzh9CzucA0axuj1Q0mgZj/3HKnZxV1LwHnoki1DnHdy4M343qJE3Gap44ArStJAKxsik1oybgqTBHhadAZhwokB393/ROygoLV0K9M+1ov+Lc1kg95N2ZPaXdyYFCHt11Dw7n6Wp8LUCNc4XJjLavghlyBDKwrcBfxHa1PPqZwnSiNhoDgLNvbqoMn6DGYHvhEjBICEFXEGStq91IH2SEg9SiCwUvm8TeuAY6nAEPjIzJOY+45Cw2WnrqxHYhg2nCM4CRc0Nrd1rre6I9Hh+7XC1N73cPJutAS2dvt3ZeAZLjb+V++HdnTl/6ZtejCwMb8LRIAoyTKQjQb2V3trJx1rZ+ugf/NN8BU370y7oM91NLLp1E8nrRakK2kDLzHmkobeBfqhz/usc/aUve6BMR4dTfggq/MiFn7a6oKRTU085LZRSgagw6ME="
eee_w_3390782@runweb141609:~$
```

### 4. Go to your EC2 instance and check for any running or existing instances, here in the photo there are no running instances.

Name:- Arnav .S. Sawant

D15A

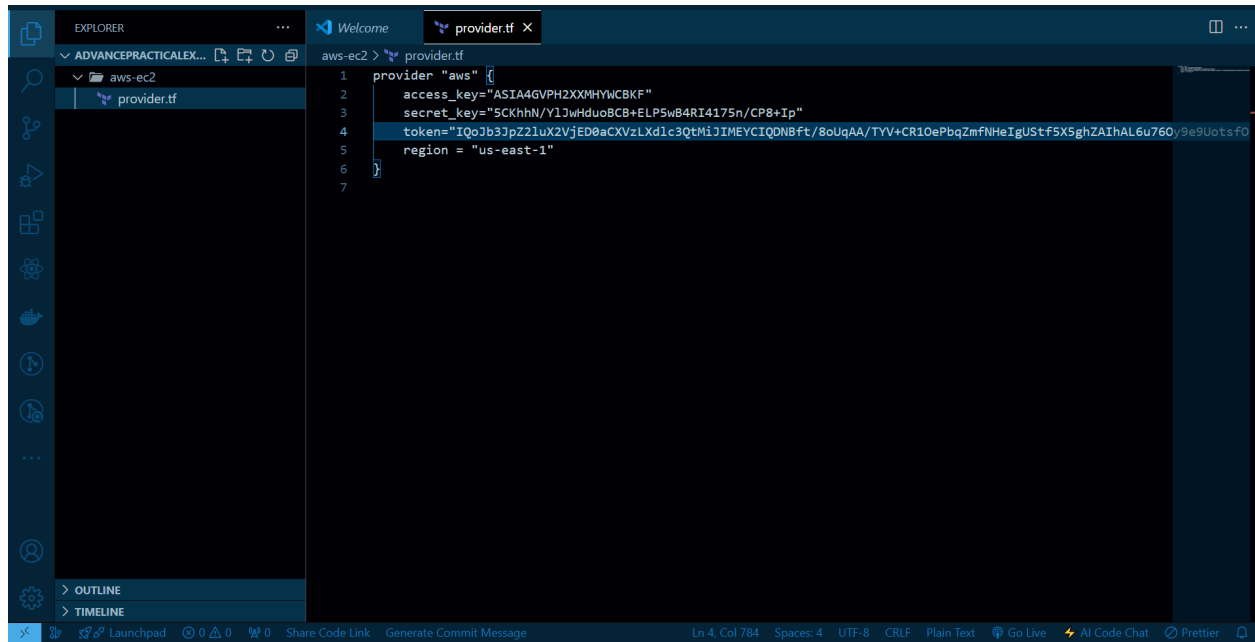
Roll-no:- 53

The screenshot displays the AWS Management Console's EC2 Dashboard. The left-hand navigation pane includes links to 'EC2 Dashboard', 'EC2 Global View', 'Events', 'Instances', 'Instance Types', 'Launch Templates', 'Spot Requests', 'Savings Plans', 'Reserved Instances', 'Dedicated Hosts', 'Capacity Reservations', 'Images', 'AMIs', and 'AMI Catalog'. The main area is titled 'Resources' and shows a summary of EC2 resources in the 'US East (N. Virginia)' region. A table lists various resource types and their counts: Instances (0), Capacity Reservations (0), Elastic IPs (0), Key pairs (2), Placement groups (0), Snapshots (0), Auto Scaling Groups (0), Dedicated Hosts (0), Instances (0), Load balancers (0), Security groups (10), and Volumes (0). Below this table are sections for 'Launch instance' (with a brief description and a 'Launch instance' button) and 'Service health' (with an 'AWS Health Dashboard' link). On the right, there are sections for 'Account attributes' (including 'Default VPC' and 'Settings') and 'Explore AWS' (with a link to 'Enable Best Price-Performance with AWS Graviton2').

5. Copy the AMI ID in the notepad which is required for further use

The screenshot shows the 'Launch instance' wizard in the AWS Management Console. The 'Summary' tab is active, displaying the following configuration: 'Number of instances' set to 1, 'Software Image (AMI)' as 'Canonical, Ubuntu, 24.04, amd64...read more' (AMI ID: ami-0866a3c8686eae6a), 'Virtual server type (instance type)' as 't2.micro', 'Firewall (security group)' as 'New security group', and 'Storage (volumes)' as '1 volume(s) - 8 GiB'. At the bottom, there are 'Cancel' and 'Launch instance' buttons. The 'Launch instance' button is orange and prominent. The 'AMI ID' field is highlighted, indicating where the user should copy the ID for further use.

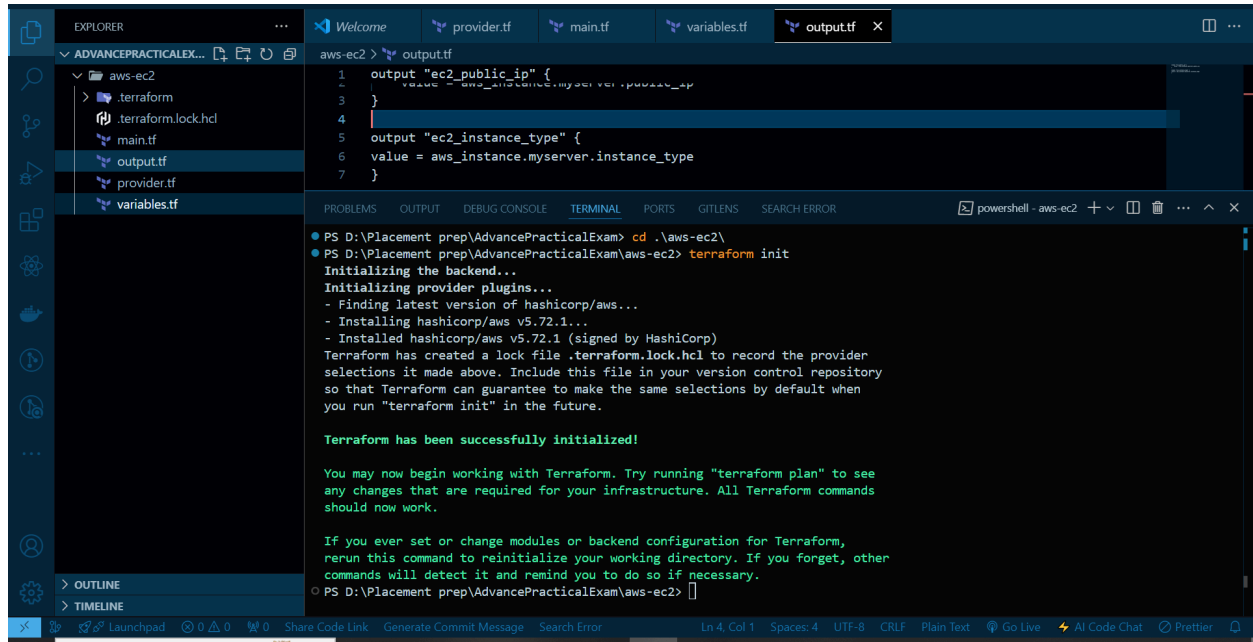
6. Paste the content of the notepad in provider.tf of the aws-ec2 folder.



The screenshot shows a Visual Studio Code editor window with a file named `provider.tf` open. The file contains Terraform configuration for the AWS provider. The Explorer sidebar on the left shows the project structure with a folder named `aws-ec2` containing the `provider.tf` file. The Outline and Timeline panels are also visible at the bottom left. The status bar at the bottom indicates the current line and column (Ln 4, Col 784) and provides links for Launchpad, Share Code Link, and Generate Commit Message.

```
1 provider "aws" {  
2   access_key = "ASIA4GVPH2XXMHYVCBKF"  
3   secret_key = "5CKhhN/Y1JwHduo8CB+ELP5wB4RI4175n/CP8+Ip"  
4   token = "IqoJb3JpZ21uX2VjED0aCXVzLXd1c3QtMiJIMEYCIQDN8ft/8oUqAA/TYV+CR10ePbqZmfNHeIgUStf5X5ghZAIhAL6u760y9e9Uotsf0"  
5   region = "us-east-1"  
6 }  
7
```

7. Write the code of `output.tf` and `variables.tf` in the respective file and on terminal write "terraform init".



```
1  output "ec2_public_ip" {
2    value = aws_instance.myserver.public_ip
3  }
4
5  output "ec2_instance_type" {
6    value = aws_instance.myserver.instance_type
7  }
```

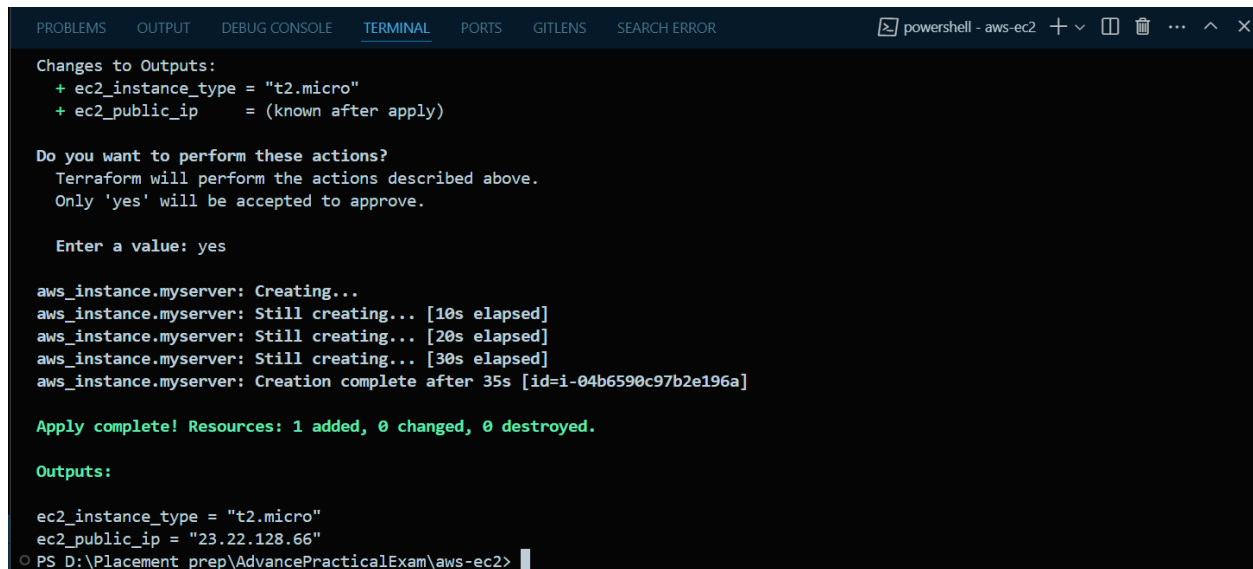
```
PS D:\Placement prep\AdvancePracticalExam> cd .\aws-ec2\
PS D:\Placement prep\AdvancePracticalExam\aws-ec2> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.72.1...
- Installed hashicorp/aws v5.72.1 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS D:\Placement prep\AdvancePracticalExam\aws-ec2>
```

8. Then execute the command “terraform plan”, check for errors, if there are no errors go for “terraform apply”



```
Changes to Outputs:
+ ec2_instance_type = "t2.micro"
+ ec2_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.myserver: Creating...
aws_instance.myserver: Still creating... [10s elapsed]
aws_instance.myserver: Still creating... [20s elapsed]
aws_instance.myserver: Still creating... [30s elapsed]
aws_instance.myserver: Creation complete after 35s [id=i-04b6590c97b2e196a]

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

Outputs:

ec2_instance_type = "t2.micro"
ec2_public_ip     = "23.22.128.66"
PS D:\Placement prep\AdvancePracticalExam\aws-ec2>
```

9. Now you can see that your ec2-instance has been created



Name:- Arnav .S. Sawant

D15A

Roll-no:- 53

The screenshot shows the AWS Management Console for the 'N. Virginia' region. The left sidebar lists navigation options like EC2 Dashboard, EC2 Global View, Events, and Instances. The main content area shows 'Instances (1/1)' with a table listing the instance 'sample server ad-dev' (ID: i-04b6590c97b2e196a) in a 'Running' state. Below the table, the 'Details' tab is active, displaying the instance summary, including the Instance ID, Public IPv4 address (23.22.128.66), Private IPv4 addresses (172.31.23.14), and Instance state (Running).

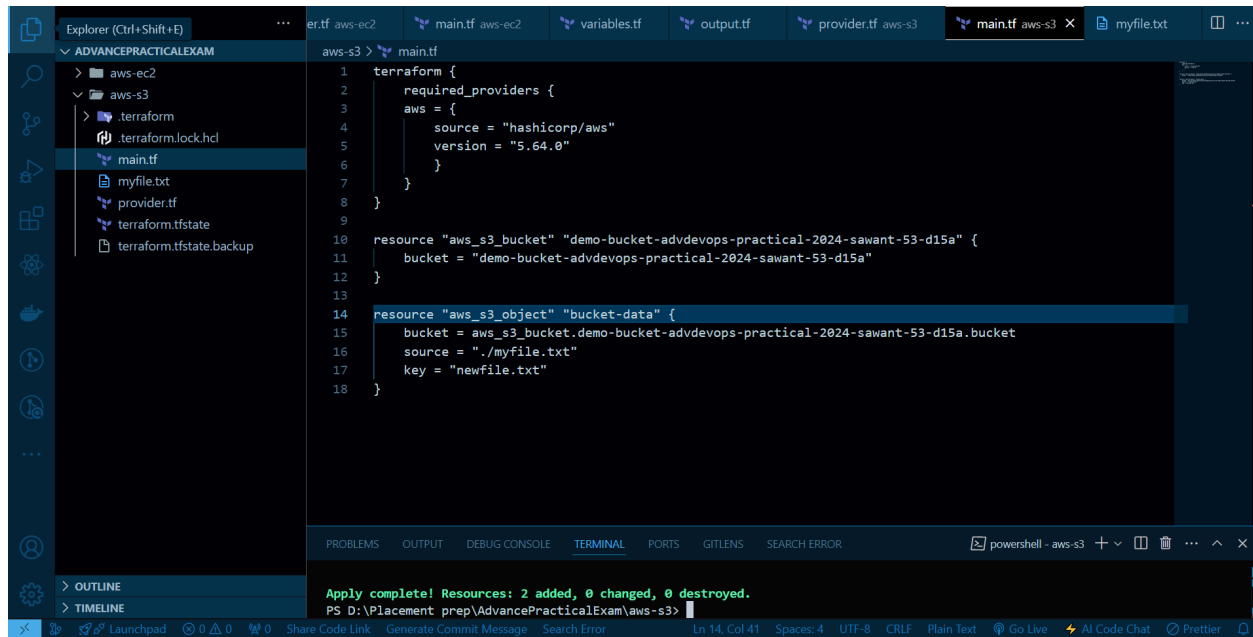
Now the part of s3

10. These are the list of s3 buckets before writing the terraform script to create an s3 bucket

The screenshot shows the AWS Management Console for the 'N. Virginia' region, specifically the 'Amazon S3' section. The 'General purpose buckets (4)' page is displayed, showing a list of buckets. The buckets are:

Name	AWS Region	IAM Access Analyzer	Creation date
<a href="#">demo-bucket-90eeef8439485e32</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	August 24, 2024, 14:43:40 (UTC+05:30)
<a href="#">elasticbeanstalk-us-east-1-838959879662</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	August 19, 2024, 17:21:21 (UTC+05:30)
<a href="#">lambdabucketarnav</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	October 8, 2024, 15:22:01 (UTC+05:30)
<a href="#">mywebapp-bucket-9b775af4d39a8e8d</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	August 24, 2024, 16:22:01 (UTC+05:30)

11. Write the code in “main.tf” for bucket and remember to make the name of the bucket globally unique

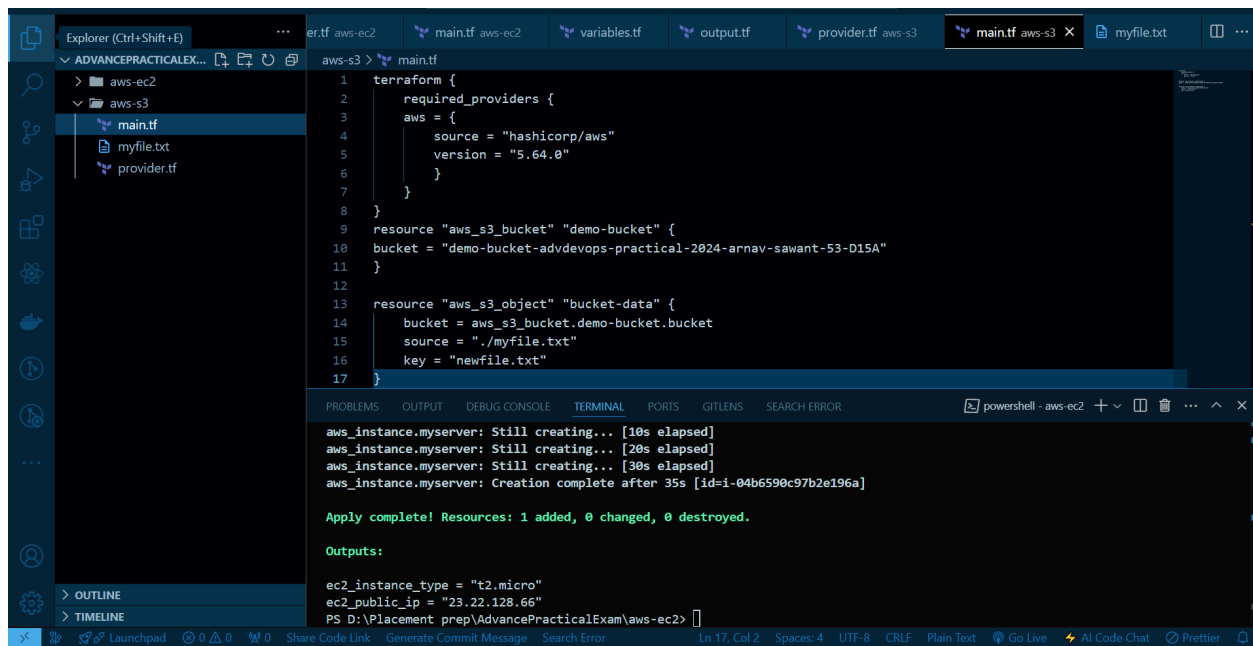


The screenshot shows the VS Code interface with the Explorer on the left displaying a project structure for 'ADVANCEPRACTICALEXAM'. The 'main.tf' file is selected. The main editor shows the following Terraform code:

```
1 terraform {
2   required_providers {
3     aws = {
4       source = "hashicorp/aws"
5       version = "5.64.0"
6     }
7   }
8 }
9
10 resource "aws_s3_bucket" "demo-bucket-advdevops-practical-2024-sawant-53-d15a" {
11   bucket = "demo-bucket-advdevops-practical-2024-sawant-53-d15a"
12 }
13
14 resource "aws_s3_object" "bucket-data" {
15   bucket = aws_s3_bucket.demo-bucket-advdevops-practical-2024-sawant-53-d15a.bucket
16   source = "../myfile.txt"
17   key = "newfile.txt"
18 }
```

The terminal at the bottom shows the command 'terraform apply' being executed, resulting in the message: 'Apply complete! Resources: 2 added, 0 changed, 0 destroyed.'

12. Now after the “terraform apply” command for EC2 the public ip is been listed in the terminal, so copy the ip address and paste in the .txt file created in the aws-s3 folder



The screenshot shows the VS Code interface with the Explorer on the left displaying a project structure for 'ADVANCEPRACTICALEXAM'. The 'main.tf' file is selected. The main editor shows the following Terraform code:

```
1 terraform {
2   required_providers {
3     aws = {
4       source = "hashicorp/aws"
5       version = "5.64.0"
6     }
7   }
8 }
9
10 resource "aws_s3_bucket" "demo-bucket" {
11   bucket = "demo-bucket-advdevops-practical-2024-arnav-sawant-53-D15A"
12 }
13
14 resource "aws_s3_object" "bucket-data" {
15   bucket = aws_s3_bucket.demo-bucket.bucket
16   source = "../myfile.txt"
17   key = "newfile.txt"
18 }
```

The terminal at the bottom shows the command 'terraform apply' being executed, resulting in the message: 'Apply complete! Resources: 1 added, 0 changed, 0 destroyed.'

Outputs:

```
ec2_instance_type = "t2.micro"
ec2_public_ip = "23.22.128.66"
```

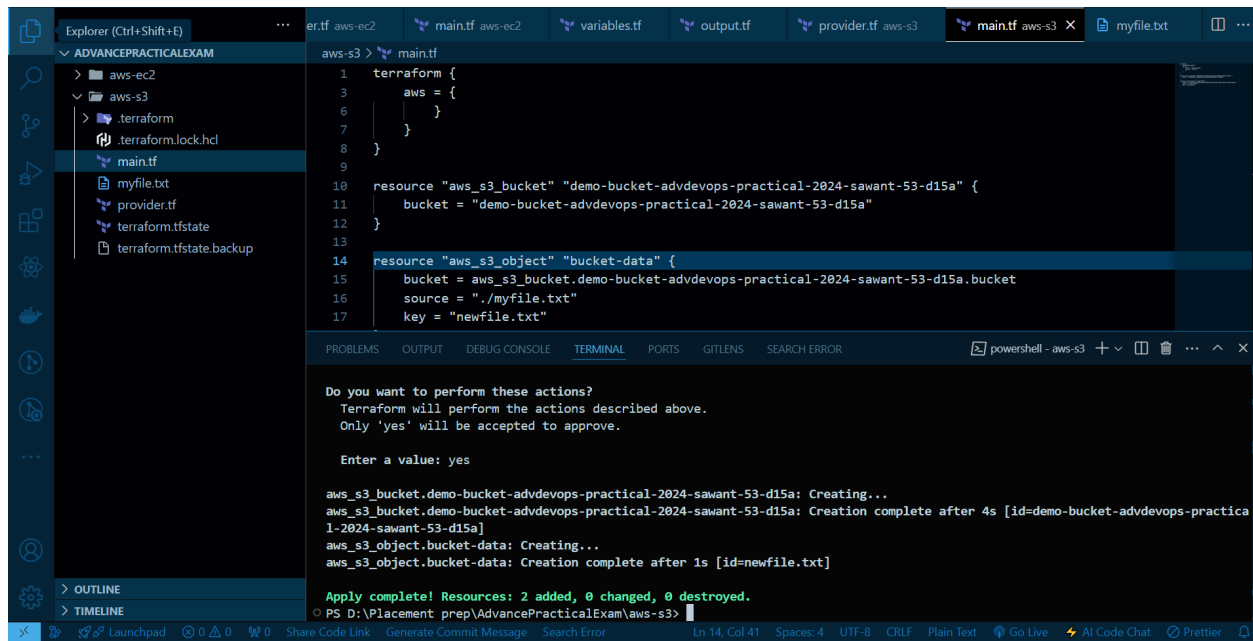
PS D:\Placement prep\AdvancePracticalExam\aws-ec2>

Name:- Arnav .S. Sawant

D15A

Roll-no:- 53

13. Same go for all the terraform commands and lastly do “terraform apply”



The screenshot shows a VS Code editor with a Terraform configuration file named `main.tf`. The configuration defines an AWS S3 bucket and an S3 object. The terminal window shows the output of the `terraform apply` command, indicating that the resources were created successfully.

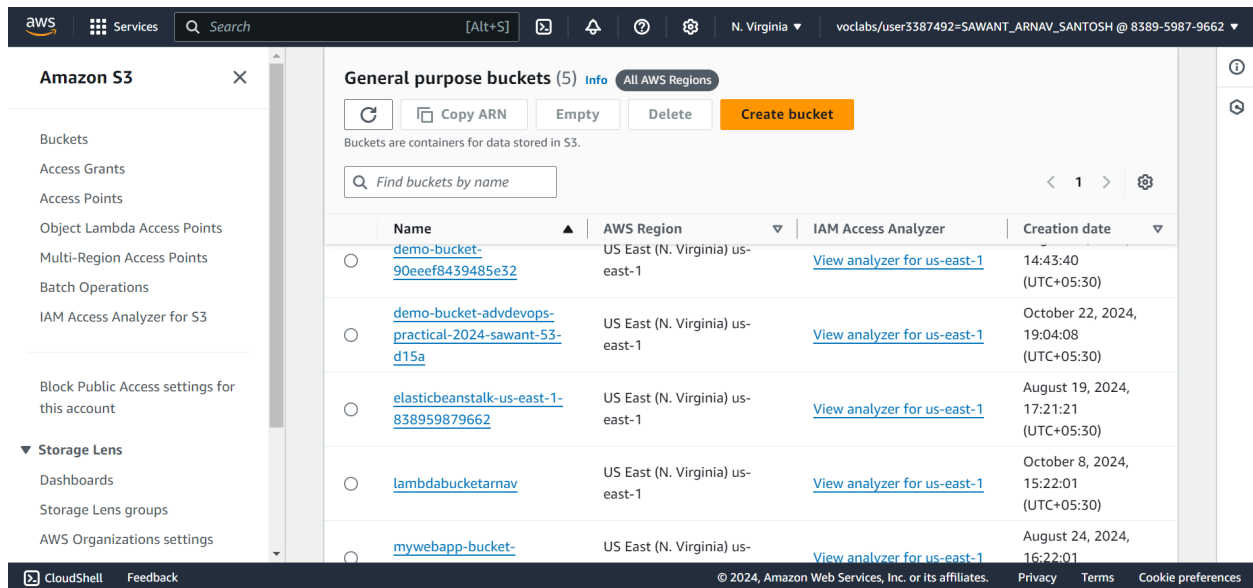
```
1 terraform {
2   aws = {
3     profile = "arnav"
4   }
5 }
6
7 resource "aws_s3_bucket" "demo-bucket-advdevops-practical-2024-sawant-53-d15a" {
8   bucket = "demo-bucket-advdevops-practical-2024-sawant-53-d15a"
9 }
10
11 resource "aws_s3_object" "bucket-data" {
12   bucket = aws_s3_bucket.demo-bucket-advdevops-practical-2024-sawant-53-d15a.bucket
13   source = "../myfile.txt"
14   key = "newfile.txt"
15 }
```

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\_s3\_bucket.demo-bucket-advdevops-practical-2024-sawant-53-d15a: Creating...  
aws\_s3\_bucket.demo-bucket-advdevops-practical-2024-sawant-53-d15a: Creation complete after 4s [id=demo-bucket-advdevops-practical-2024-sawant-53-d15a]  
aws\_s3\_object.bucket-data: Creating...  
aws\_s3\_object.bucket-data: Creation complete after 1s [id=newfile.txt]

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

14. Now after doing that you can see that the bucket has been created successfully



The screenshot shows the AWS Management Console for the 'General purpose buckets' section. The table lists the buckets created, including the one specified in the Terraform configuration.

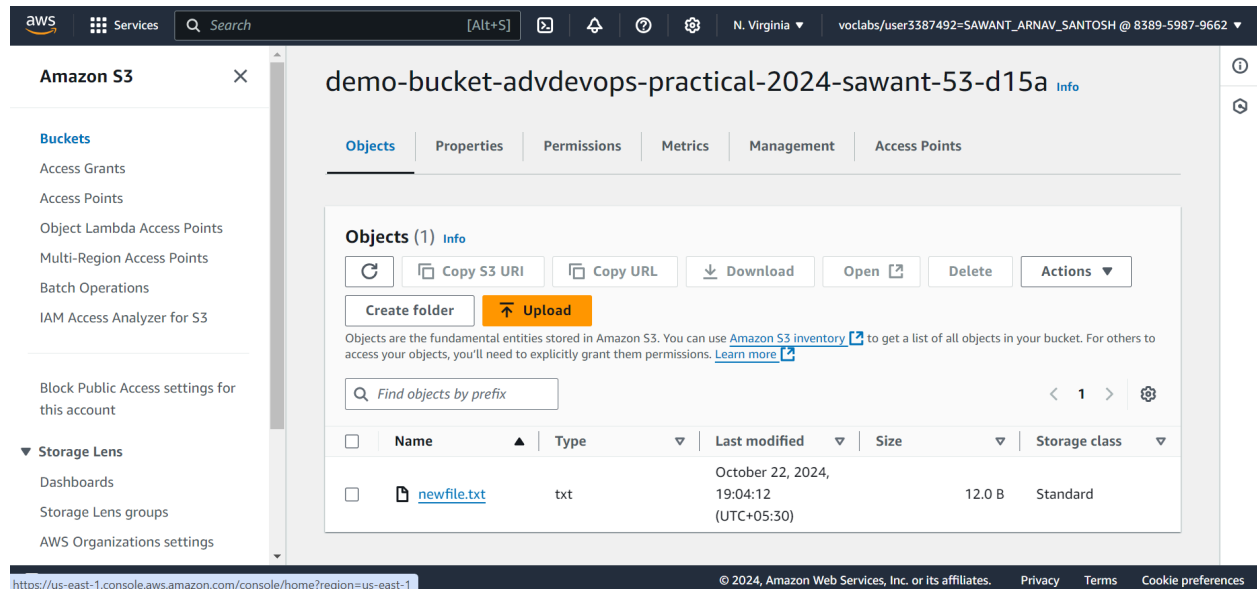
Name	AWS Region	IAM Access Analyzer	Creation date
<a href="#">demo-bucket-90eef8439485e32</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	14:43:40 (UTC+05:30)
<a href="#">demo-bucket-advdevops-practical-2024-sawant-53-d15a</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	October 22, 2024, 19:04:08 (UTC+05:30)
<a href="#">elasticbeanstalk-us-east-1-838959879662</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	August 19, 2024, 17:21:21 (UTC+05:30)
<a href="#">lambdabucketarnav</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	October 8, 2024, 15:22:01 (UTC+05:30)
<a href="#">mywebapp-bucket-</a>	US East (N. Virginia) us-	<a href="#">View analyzer for us-east-1</a>	August 24, 2024, 16:22:01

Name:- Arnav .S. Sawant

D15A

Roll-no:- 53

15. Also the newfile.txt has been created and the ip of the ec2 instance has been stored there



Then destroy both the ec2 instance and s3 bucket using command “terraform destroy”

For s3 -

```
PS D:\Placement prep\AdvancePracticalExam\aws-s3> terraform destroy
● aws_s3_bucket.demo-bucket-advdevops-practical-2024-sawant-53-d15a: Refreshing state... [id=demo-bucket-advdevops-practical-2024-sawant-53-d15a]
aws_s3_object.bucket-data: Refreshing state... [id=newfile.txt]

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_s3_bucket.demo-bucket-advdevops-practical-2024-sawant-53-d15a will be destroyed
- resource "aws_s3_bucket" "demo-bucket-advdevops-practical-2024-sawant-53-d15a" {
  - arn              = "arn:aws:s3:::demo-bucket-advdevops-practical-2024-sawant-53-d15a" -> null
  - bucket           = "demo-bucket-advdevops-practical-2024-sawant-53-d15a" -> null
  - bucket_domain_name = "demo-bucket-advdevops-practical-2024-sawant-53-d15a.s3.amazonaws.com" -> null
  - bucket_regional_domain_name = "demo-bucket-advdevops-practical-2024-sawant-53-d15a.s3.us-east-1.amazonaws.com" -> null
  - force_destroy     = false -> null
  - hosted_zone_id    = "Z3AQBSTGFYJSTF" -> null
```

For ec2 -

```
PS D:\Placement prep\AdvancePracticalExam\aws-s3> cd ..
PS D:\Placement prep\AdvancePracticalExam> cd .\aws-ec2\
PS D:\Placement prep\AdvancePracticalExam\aws-ec2> terraform destroy
● aws_instance.myserver: Refreshing state... [id=i-04b6590c97b2e196a]

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.myserver will be destroyed
- resource "aws_instance" "myserver" {
  - ami                        = "ami-0866a3c8686eaeaba" -> null
  - arn                      = "arn:aws:ec2:us-east-1:838959879662:instance/i-04b6590c97b2e196a" -> null
  - associate_public_ip_address = true -> null
  - availability_zone         = "us-east-1a" -> null
  - cpu_core_count            = 1 -> null
```