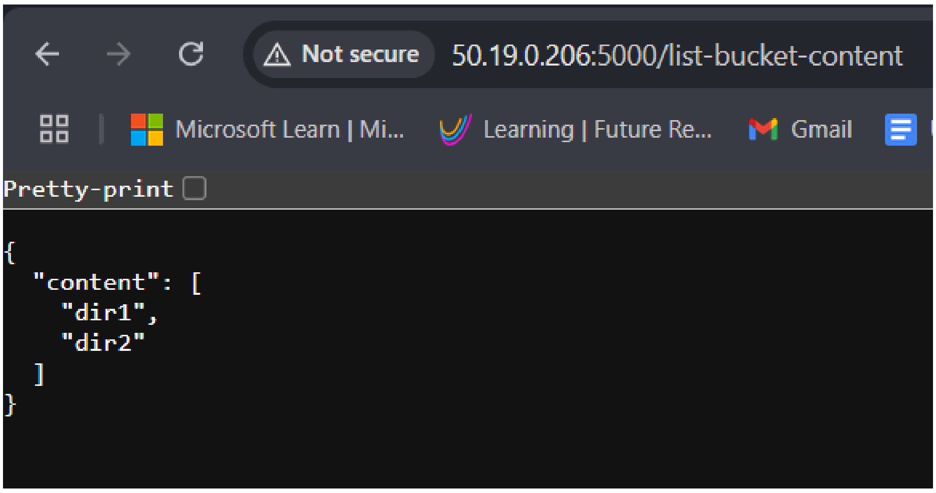
Final Output

GET http://IP:PORT/list-bucket-content

○ Response: {"content": ["dir1", "dir2"}

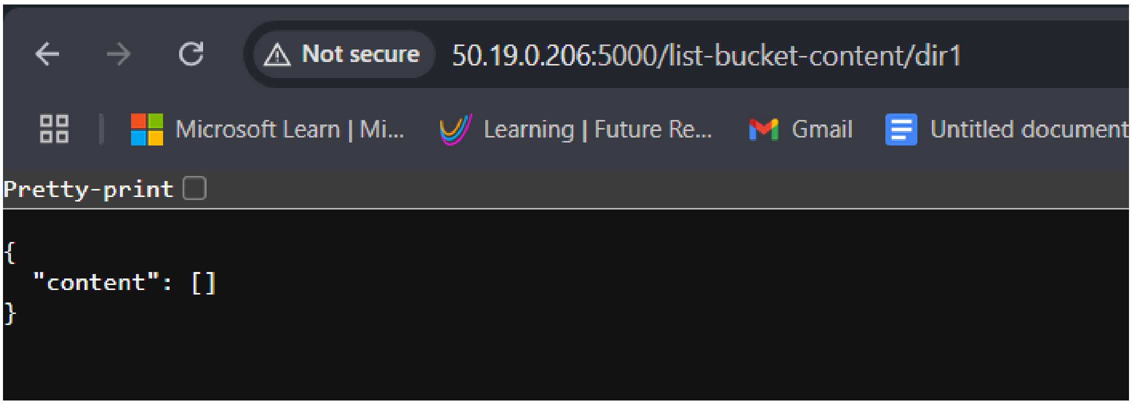


Pretty-print □

```
{
  "content": [
    "dir1",
    "dir2"
  ]
}
```

GET http://IP:PORT/list-bucket-content/dir1

○ Response: {"content": []}

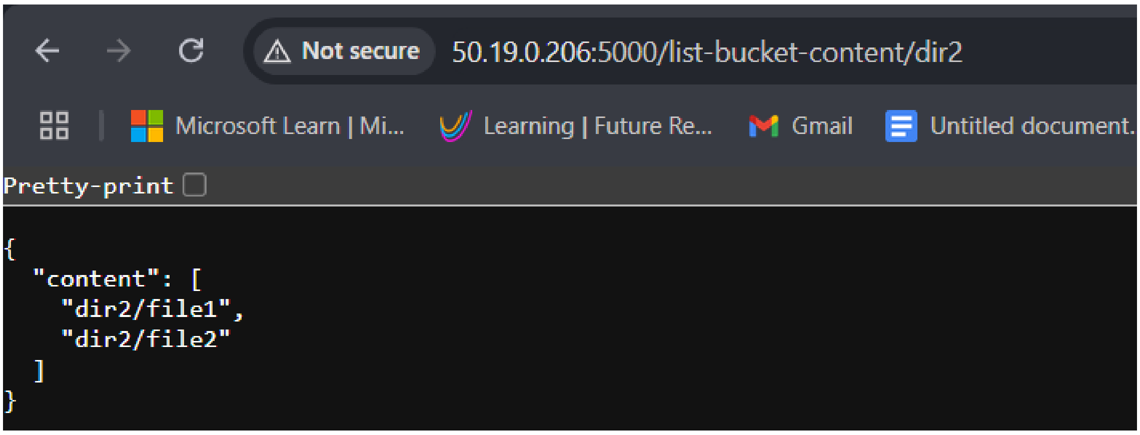


Pretty-print □

```
{
  "content": []
}
```

GET http://IP:PORT/list-bucket-content/dir2

○ Response: {"content": ["file1", "file2"]}



Pretty-print □

```
{
  "content": [
    "dir2/file1",
    "dir2/file2"
  ]
}
```

this assesment involve 2 steps

1.Developing an HTTP service using a programming language of your choice.

2.Using Terraform to deploy this service on AWS.

Prerequisites

Before proceeding i have installed the following set up:

AWS account: will need access to an AWS account with appropriate IAM permissions.

AWS CLI: Ensure we have the AWS Command Line Interface (CLI) installed and configured with the proper credentials.

Terraform: Make sure Terraform is installed on ec2 machine.

Python 3: The HTTP service is written in Python, so we need Python 3 and pip installed

Structure of the equip9 assesment i used

```
├── equip9/
│   ├── http_code/              # Contains the Flask app for the HTTP service
│   │   ├── equip_app.py        # Python Flask app for the HTTP service
│   │   └── add_files.py        # Script to add files to S3 bucket
│   └── terraform_e9/           # Contains the Terraform files for AWS deployment
│       ├── main.tf             # Terraform configuration for provisioning AWS resources
│       ├── terraform.tfstate   # Terraform state file
│       └── terraform.tfstate.backup  # Backup state file
```

Part 1: HTTP Service

Objective:

The goal of the HTTP service is to expose an endpoint that lists the contents of a specified path within an S3 bucket. If no path is specified, it lists the top-level files and directories.

Implementation:

The service is built using Flask and Boto3 (AWS SDK for Python). The service exposes the following API endpoint:

GET http://50.19.0.206:5000/list-bucket-content is list-bucket-content: Returns the top-level contents of the S3 bucket.

GET http://50.19.0.206:5000/list-bucket-content/dir1 : Lists the contents of the S3 directory specified in the <path> parameter.

Example Usage:

GET http://50.19.0.206:5000/list-bucket-content/dir2/list-bucket-content:

Response:

```
{
  "content": ["dir1", "dir2", "file1", "file2"]
}
```

GET http://50.19.0.206:5000/list-bucket-content/dir1:

```
{
  "content": []
}
```

GET http://50.19.0.206:5000/list-bucket-content/dir2:

```
{
  "content": ["file1", "file2"]
}
```

How it works:

Flask is used to create a simple API.

Boto3 is used to interact with AWS S3 to list the contents of the bucket and directories.

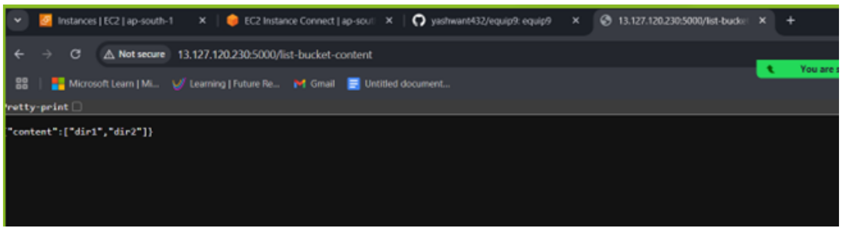
The S3 client uses list_objects_v2 to get the objects inside the bucket.

If no path is provided, it lists the top-level files and folders.

If a specific path (directory) is provided, it lists the contents of that path.

attaching screenshots of the final output

1. For http://50.19.0.206:5000/list-bucket-content


Instances | EC2 | ap-south-1
EC2 Instance Connect | ap-sout
yashwant432/equip9: equip9
13.127.120.230:5000/list-bucke
Not secure
Microsoft Learn | Mi...
Learning | Future Re...
Gmail
Untitled document...
You are s
Pretty-print □

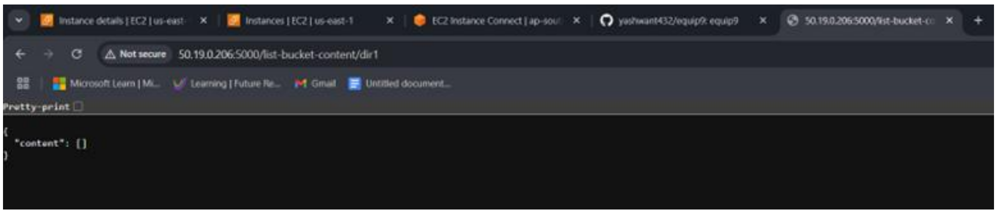
13.127.120.230:5000/list-bucket-content

```
"content":["dir1","dir2"]}
```

![alt text](image.png)

2. for http://50.19.0.206:5000/list-bucket-content/dir1

![alt text](image-1.png)

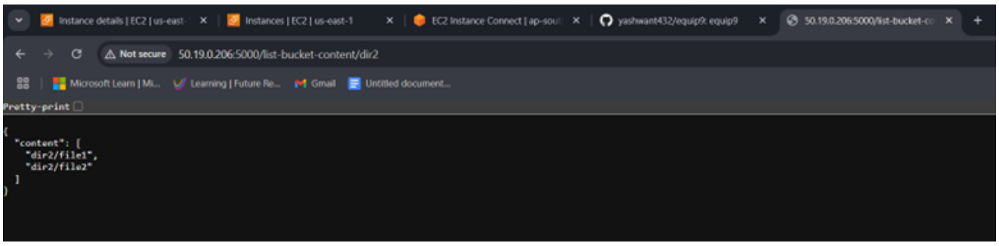

Instance details | EC2 | us-east-
Instances | EC2 | us-east-1
EC2 Instance Connect | ap-sout
yashwant432/equip9: equip9
50.19.0.206:5000/list-bucket-co
Not secure
Microsoft Learn | Mi...
Learning | Future Re...
Gmail
Untitled document...
Pretty-print □

50.19.0.206:5000/list-bucket-content/dir1

```
{
  "content": []
}
```

3. for http://50.19.0.206:5000/list-bucket-content/dir2

![alt text](image-2.png)


Instance details | EC2 | us-east-
Instances | EC2 | us-east-1
EC2 Instance Connect | ap-sout
yashwant432/equip9: equip9
50.19.0.206:5000/list-bucket-co
Not secure
Microsoft Learn | Mi...
Learning | Future Re...
Gmail
Untitled document...
Pretty-print □

50.19.0.206:5000/list-bucket-content/dir2

```
{
  "content": [
    "dir2/file1",
    "dir2/file2"
  ]
}
```

i have tested on localy befor deploying on ec2 instance

How to run locally:

Install the required dependencies:

go the the code path

```
├── equip9/
   ├── http_code/            # Contains the Flask app for the HTTP service
   ── equip_app.py   # code
```

Run the app:

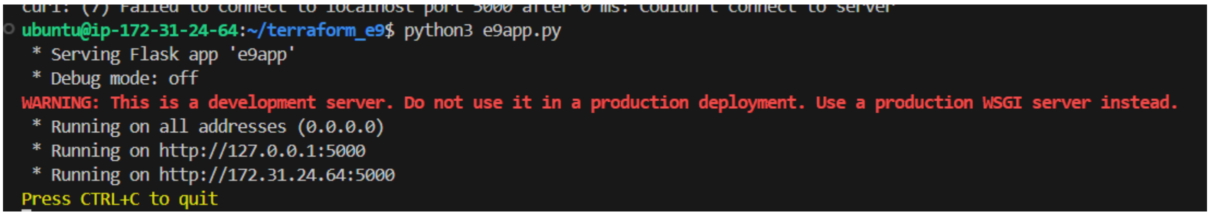
python3 equip_app.py

The Flask app will start, and you can test it using curl or Postman by visiting:

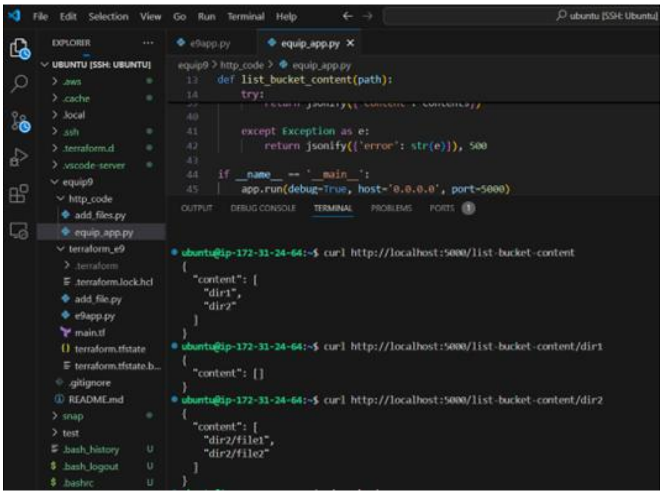
http://127.0.0.1:5000/list-bucket-content

http://127.0.0.1:5000/list-bucket-content/dir1

http://127.0.0.1:5000/list-bucket-content/dir2 attaching screenshots for local run

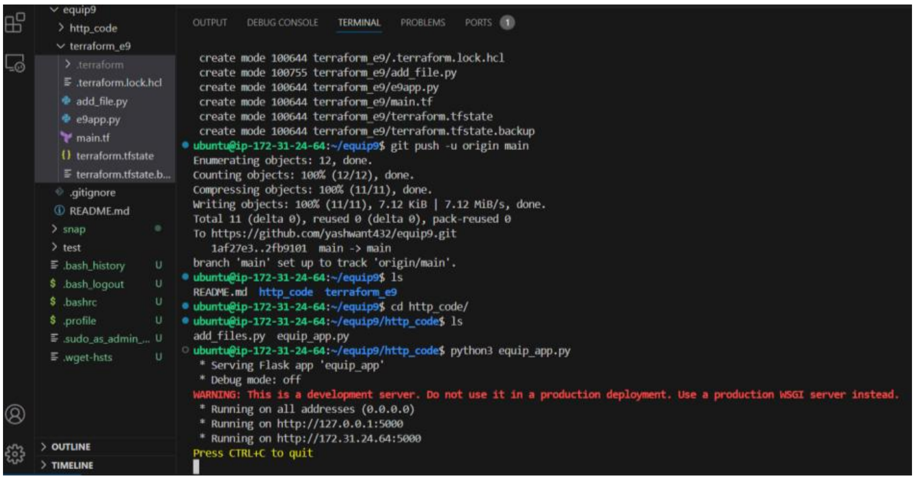


```
curl: (7) Failed to connect to localhost port 5000 after 0 ms: Couldn't connect to server
ubuntu@ip-172-31-24-64:~/terraform_e9$ python3 e9app.py
 * Serving Flask app 'e9app'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.31.24.64:5000
Press CTRL+C to quit
```



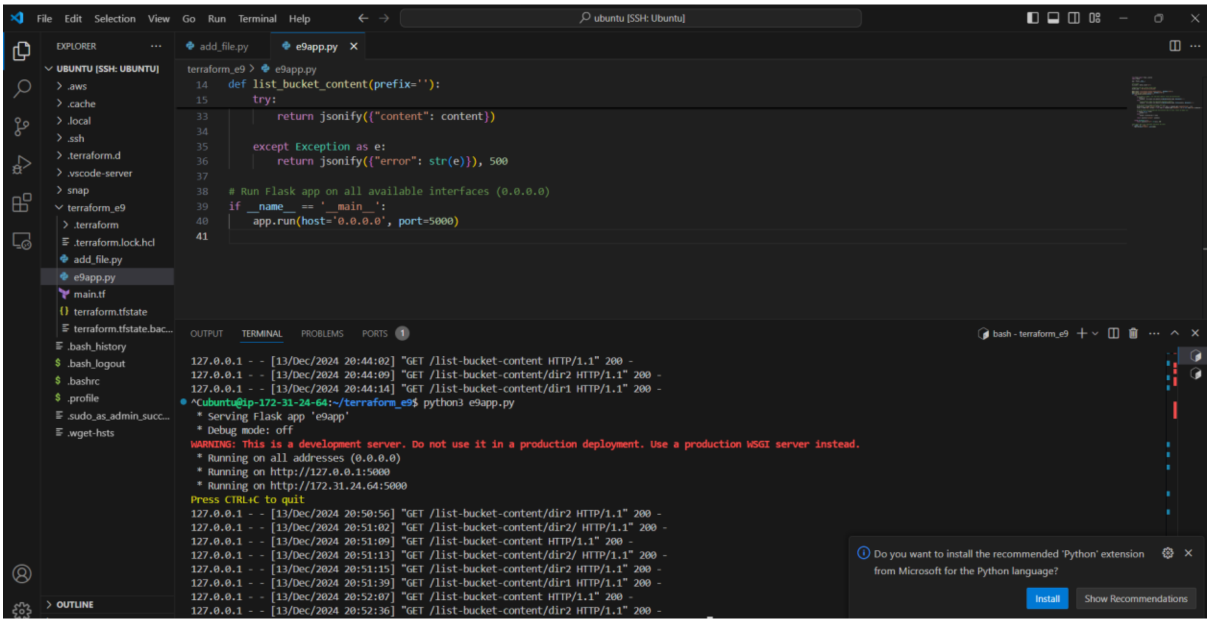
```
def list_bucket_content(path):
    try:
        return jsonify({'content': contents})

    except Exception as e:
        return jsonify({'error': str(e)}), 500

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=5000)
```

```
ubuntu@ip-172-31-24-64:~$ curl http://localhost:5000/list-bucket-content
{
  "content": [
    "dir1",
    "dir2"
  ]
}
ubuntu@ip-172-31-24-64:~$ curl http://localhost:5000/list-bucket-content/dir1
{
  "content": []
}
ubuntu@ip-172-31-24-64:~$ curl http://localhost:5000/list-bucket-content/dir2
{
  "content": [
    "dir2/file1",
    "dir2/file2"
  ]
}
```



```
 create mode 100644 terraform_e9/.terraform.lock.hcl
 create mode 100755 terraform_e9/add_file.py
 create mode 100644 terraform_e9/e9app.py
 create mode 100644 terraform_e9/main.tf
 create mode 100644 terraform_e9/terraform.tfstate
 create mode 100644 terraform_e9/terraform.tfstate.backup
ubuntu@ip-172-31-24-64:~/equip9$ git push -u origin main
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 7.12 KiB | 7.12 MiB/s, done.
Total 11 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/yashwant432/equip9.git
   1af27e3..2fb9101  main -> main
branch 'main' set up to track 'origin/main'.
ubuntu@ip-172-31-24-64:~/equip9$ ls
README.md  http_code  terraform_e9
ubuntu@ip-172-31-24-64:~/equip9$ cd http_code/
ubuntu@ip-172-31-24-64:~/equip9/http_code$ ls
add_files.py  equip_app.py
ubuntu@ip-172-31-24-64:~/equip9/http_code$ python3 equip_app.py
 * Serving Flask app 'equip_app'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.31.24.64:5000
Press CTRL+C to quit
```

![alt text](image-18.png)



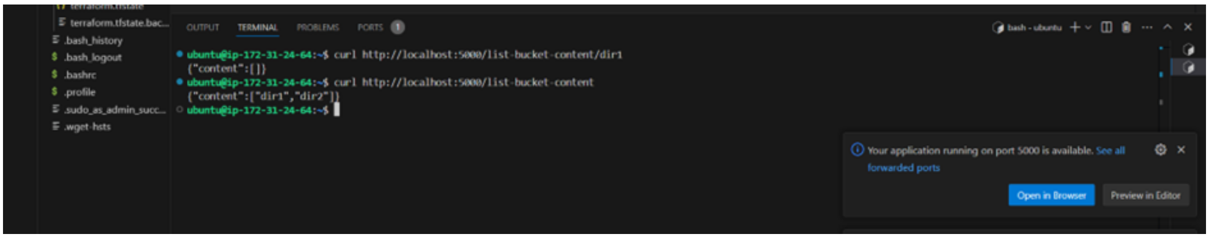
```python
def list_bucket_content(prefix=''):
    try:
            return jsonify({"content": content})

    except Exception as e:
        return jsonify({"error": str(e)}), 500

# Run Flask app on all available interfaces (0.0.0.0)
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

```
127.0.0.1 - - [13/Dec/2024 20:44:02] "GET /list-bucket-content HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2024 20:44:09] "GET /list-bucket-content/dir2 HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2024 20:44:14] "GET /list-bucket-content/dir1 HTTP/1.1" 200 -
^Cubuntu@ip-172-31-24-64:~/terraform_e9$ python3 e9app.py
 * Serving Flask app 'e9app'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.31.24.64:5000
Press CTRL+C to quit
127.0.0.1 - - [13/Dec/2024 20:50:56] "GET /list-bucket-content/dir2 HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2024 20:51:02] "GET /list-bucket-content/dir2/ HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2024 20:51:09] "GET /list-bucket-content HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2024 20:51:13] "GET /list-bucket-content/dir2/ HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2024 20:51:15] "GET /list-bucket-content/dir2 HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2024 20:51:39] "GET /list-bucket-content/dir1 HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2024 20:52:07] "GET /list-bucket-content HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2024 20:52:36] "GET /list-bucket-content/dir2 HTTP/1.1" 200 -
```

Do you want to install the recommended 'Python' extension from Microsoft for the Python language?

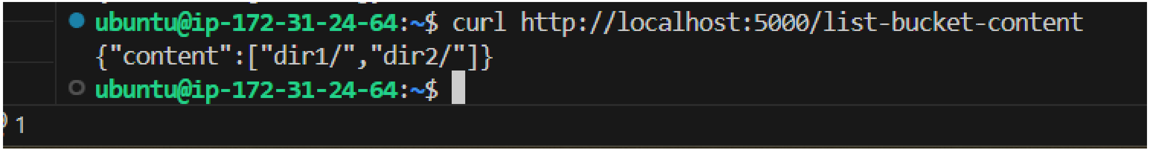
output screenshot

![alt text](image-3.png)



```
ubuntu@ip-172-31-24-64:~$ curl http://localhost:5000/list-bucket-content/dir1
{"content":[]}
ubuntu@ip-172-31-24-64:~$ curl http://localhost:5000/list-bucket-content
{"content":["dir1","dir2"]}
ubuntu@ip-172-31-24-64:~$
```

Your application running on port 5000 is available. See all forwarded ports



```
ubuntu@ip-172-31-24-64:~$ curl http://localhost:5000/list-bucket-content
{"content":["dir1/","dir2/"]}
ubuntu@ip-172-31-24-64:~$
```

Part 2: Terraform Deployment

Objective:

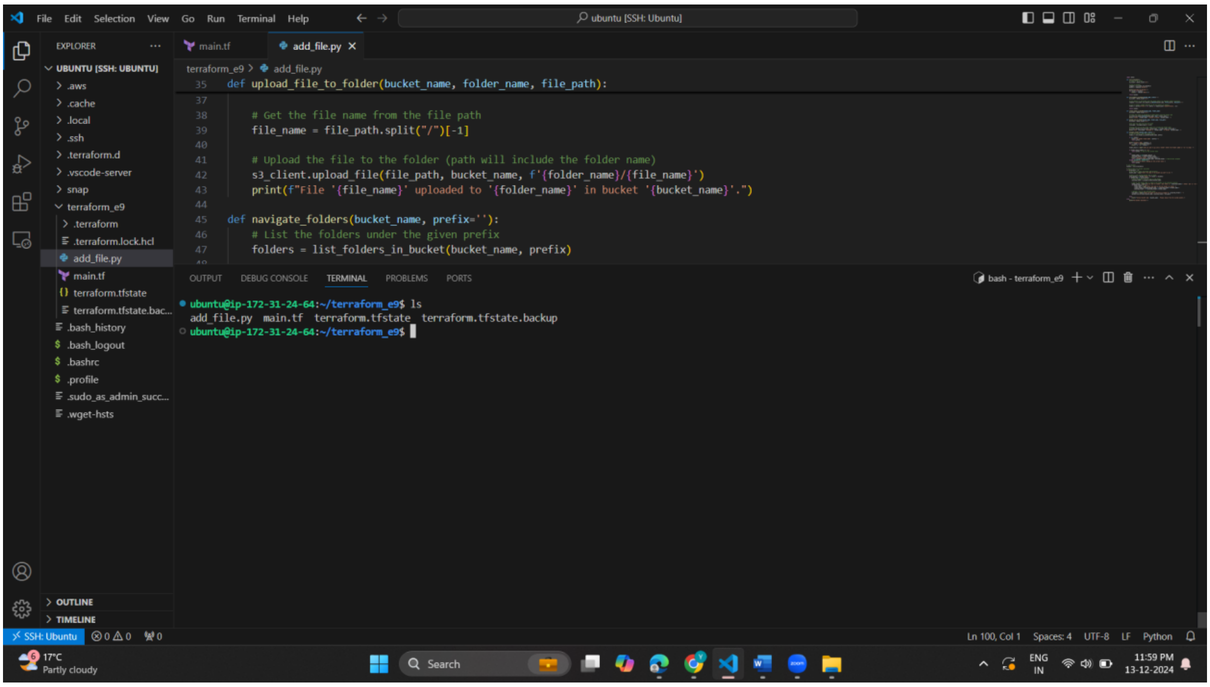
The goal of this part is to automate the deployment of the HTTP service using Terraform. Terraform is used to provision AWS infrastructure, including an EC2 instance to run the Flask application and an S3 bucket to store the files.

Steps to Deploy:

Configure AWS Credentials: Ensure your AWS CLI is configured with your credentials. You can configure it by running:

aws configure

Set up Terraform:

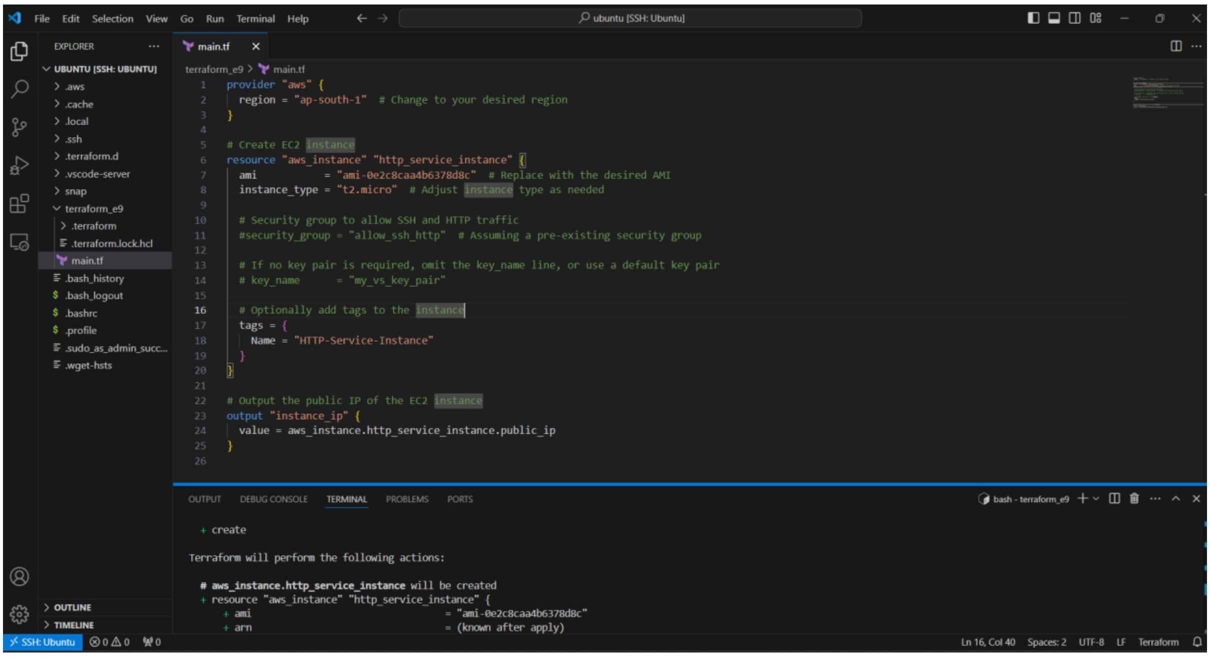


```python
def upload_file_to_folder(bucket_name, folder_name, file_path):

    # Get the file name from the file path
    file_name = file_path.split("/")[-1]

    # Upload the file to the folder (path will include the folder name)
    s3_client.upload_file(file_path, bucket_name, f'{folder_name}/{file_name}')
    print(f"File '{file_name}' uploaded to '{folder_name}' in bucket '{bucket_name}'.")

def navigate_folders(bucket_name, prefix=''):
    # List the folders under the given prefix
    folders = list_folders_in_bucket(bucket_name, prefix)
```

```
ubuntu@ip-172-31-24-64:~/terraform_e9$ ls
add_file.py  main.tf  terraform.tfstate  terraform.tfstate.backup
ubuntu@ip-172-31-24-64:~/terraform_e9$
```

Install Terraform as was not installed yet.

Created main.tf file, which will:



```hcl
provider "aws" {
  region = "ap-south-1"  # Change to your desired region
}

# Create EC2 instance
resource "aws_instance" "http_service_instance" {
  ami           = "ami-0e2c8caa4b6378d8c"  # Replace with the desired AMI
  instance_type = "t2.micro"  # Adjust instance type as needed

  # Security group to allow SSH and HTTP traffic
  #security_group = "allow_ssh_http"  # Assuming a pre-existing security group

  # If no key pair is required, omit the key_name line, or use a default key pair
  # key_name      = "my_vs_key_pair"

  # Optionally add tags to the instance
  tags = {
    Name = "HTTP-Service-Instance"
  }
}

# Output the public IP of the EC2 instance
output "instance_ip" {
  value = aws_instance.http_service_instance.public_ip
}
```

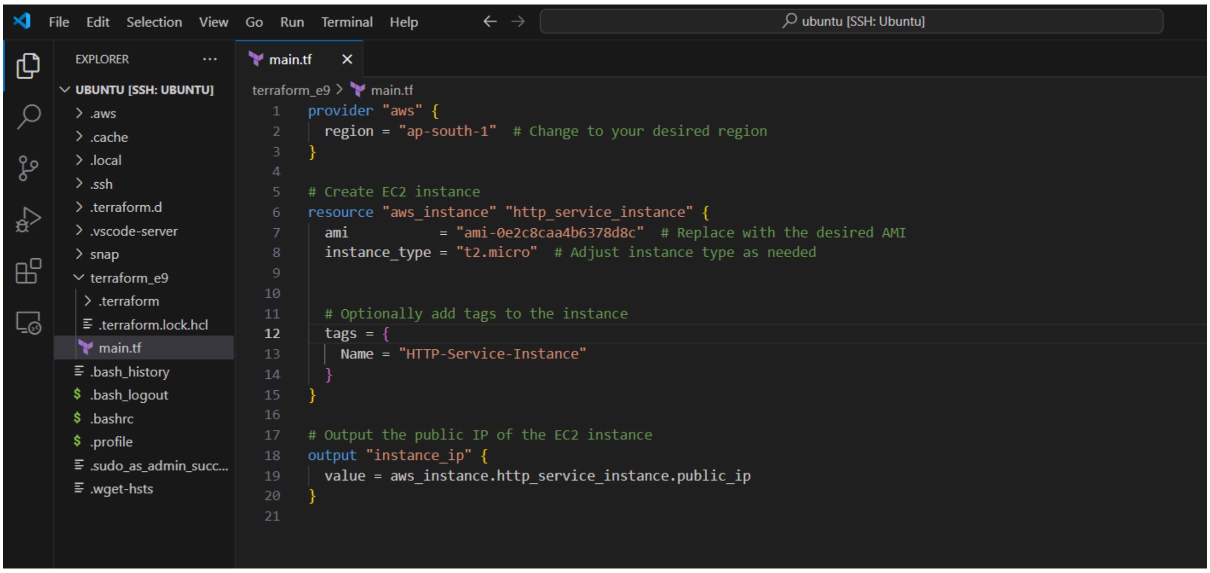
```
  + create

Terraform will perform the following actions:

  # aws_instance.http_service_instance will be created
  + resource "aws_instance" "http_service_instance" {
      + ami                                  = "ami-0e2c8caa4b6378d8c"
      + arn                                  = (known after apply)
```

Provision an EC2 instance that will host the Flask app.

Create an S3 bucket to store the files.

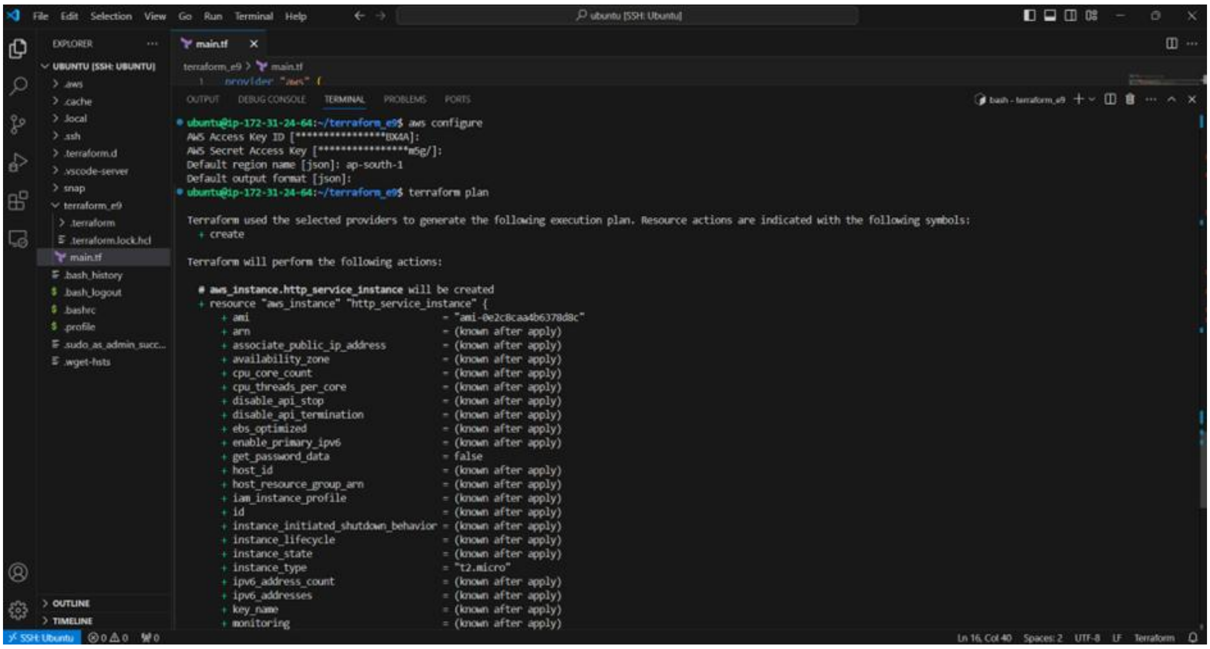
main.tf configuration:

main.tf



```
provider "aws" {
  region = "ap-south-1"  # Change to your desired region
}

# Create EC2 instance
resource "aws_instance" "http_service_instance" {
  ami           = "ami-0e2c8caa4b6378d8c"  # Replace with the desired AMI
  instance_type = "t2.micro"  # Adjust instance type as needed


  # Optionally add tags to the instance
  tags = {
    Name = "HTTP-Service-Instance"
  }
}

# Output the public IP of the EC2 instance
output "instance_ip" {
  value = aws_instance.http_service_instance.public_ip
}
```

after creating the main.tf Deploy with Terraform:

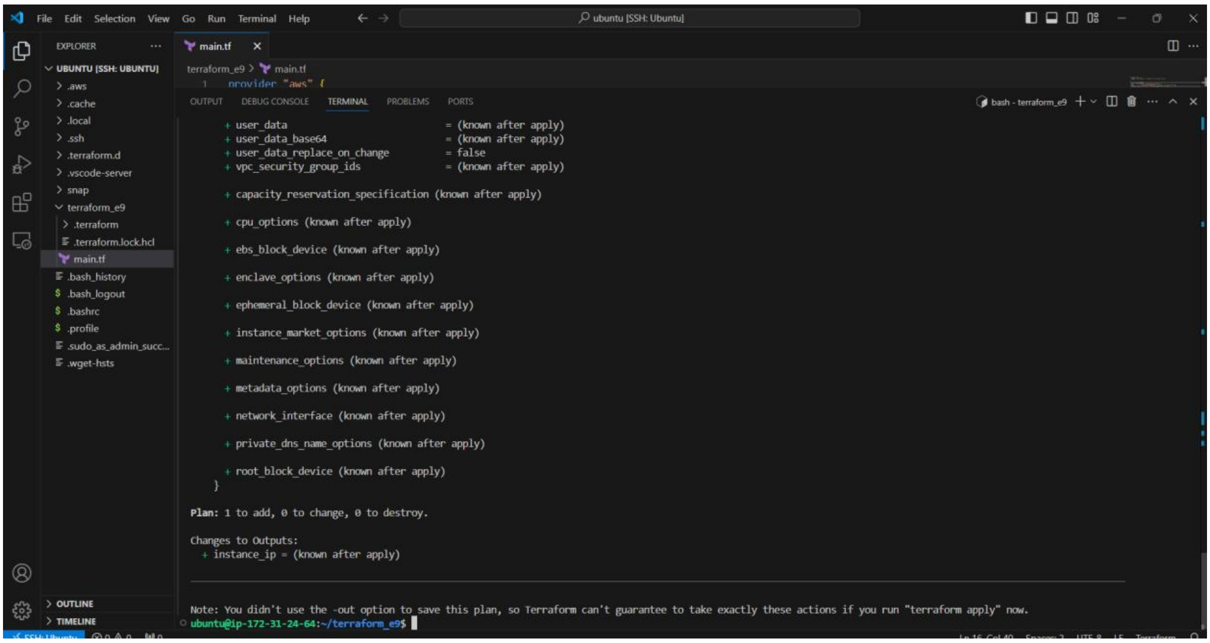


```
ubuntu@ip-172-31-24-64:~/terraform_e9$ aws configure
AWS Access Key ID [****************BX4A]:
AWS Secret Access Key [****************m5g/]:
Default region name [json]: ap-south-1
Default output format [json]:
ubuntu@ip-172-31-24-64:~/terraform_e9$ terraform plan

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.http_service_instance will be created
  + resource "aws_instance" "http_service_instance" {
      + ami                                  = "ami-0e2c8caa4b6378d8c"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + 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                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
```

Initialize Terraform:

terraform init

plan the configuration to check :terraform plan



```
provider "aws" {
```

```
      + user_data                            = (known after apply)
      + 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:
  + instance_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.
ubuntu@ip-172-31-24-64:~/terraform_e9$
```

Apply the configuration:

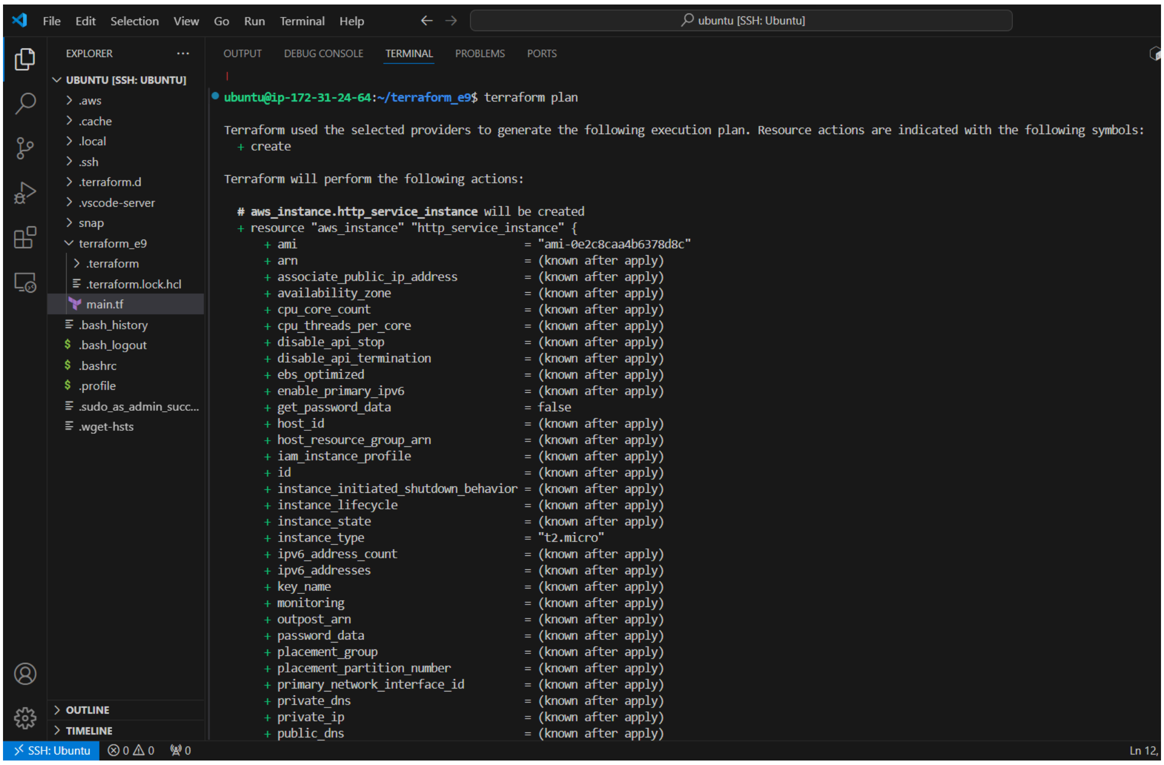
terraform apply

will create the necessary resources in AWS, including the EC2 instance and the S3 bucket.

attaching screenshots but initialization, plan and apply step in recorded video

terraform destroy

terraform plan



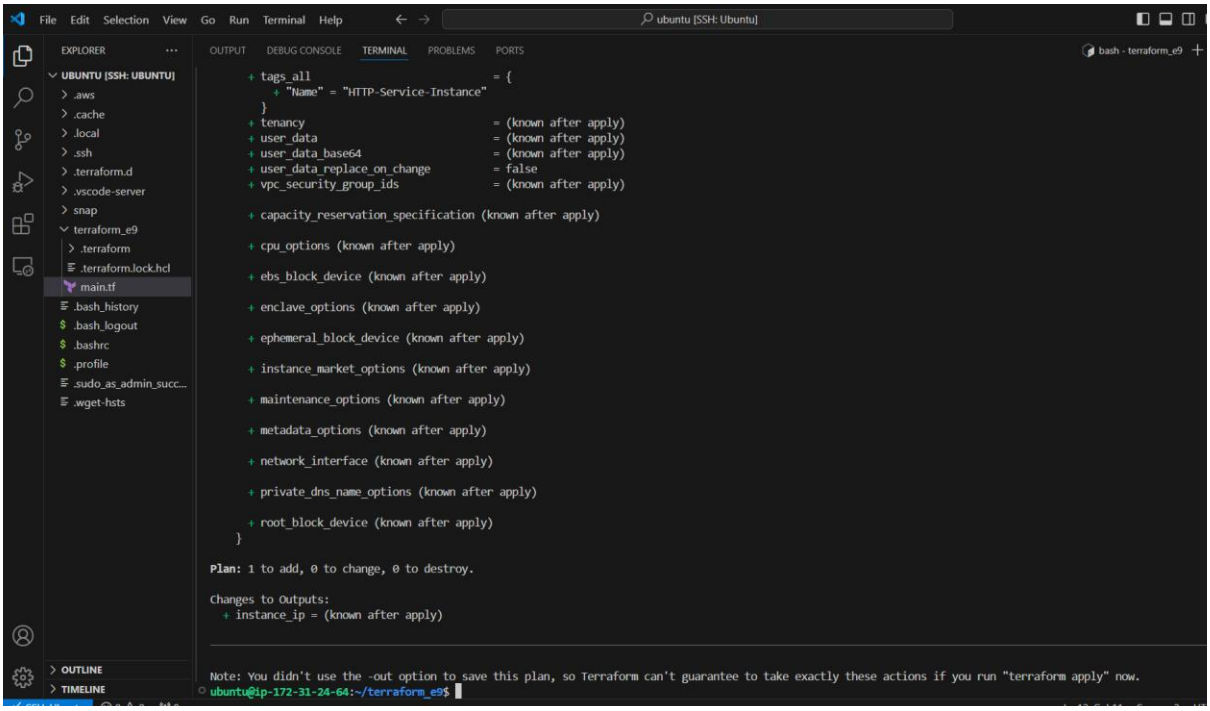
```
ubuntu@ip-172-31-24-64:~/terraform_e9$ terraform plan

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.http_service_instance will be created
  + resource "aws_instance" "http_service_instance" {
      + ami                                  = "ami-0e2c8caa4b6378d8c"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + 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                        = "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)
      + public_dns                           = (known after apply)
```

after terraform apply

![alt text](image-9.png)   - resource added by aaply

![alt text](image-8.png)   - s3 bucket creation

![alt text](image-10.png) ![alt text](image-11.png) - creation of instance



```
      + tags_all                             = {
          + "Name" = "HTTP-Service-Instance"
        }
      + tenancy                              = (known after apply)
      + user_data                            = (known after apply)
      + 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:
  + instance_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.
ubuntu@ip-172-31-24-64:~/terraform_e9$
```

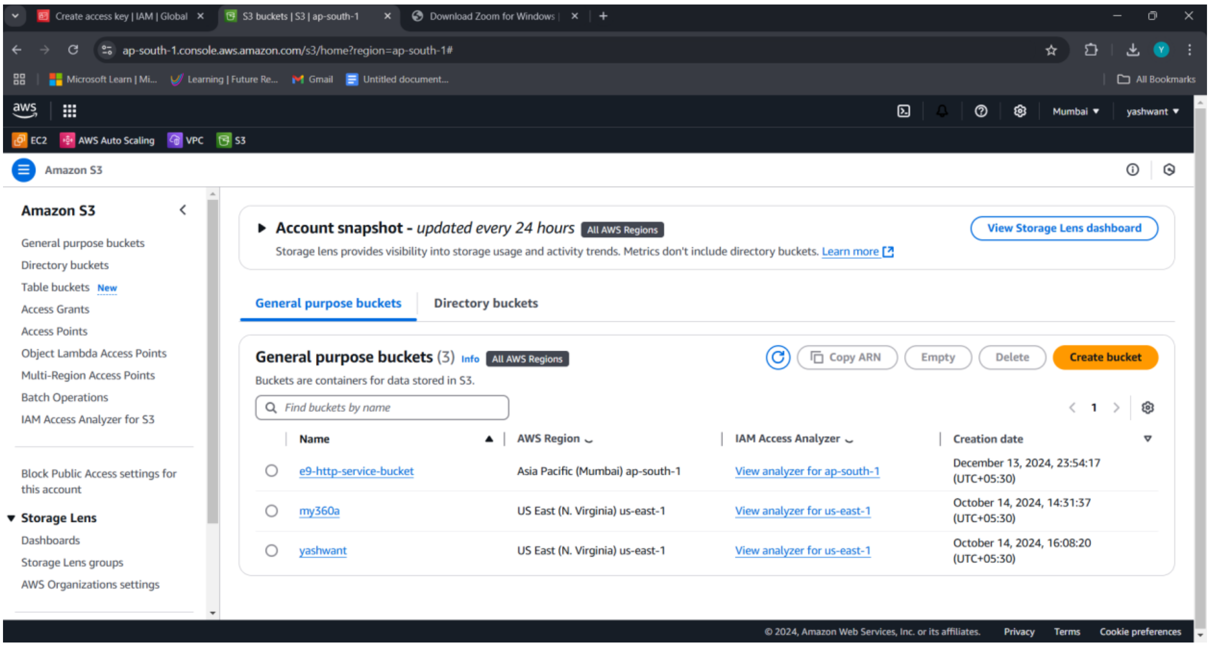
Folder creation in bucket using python

![alt text](image-12.png) - dir1 and dir2

![alt text](image-13.png)   ![alt text](image-14.png) -file1 and file2

output foe dir1 and dir 2 ![alt text](image-15.png) ![alt text](image-16.png)

s3 bucket url s3://e9-http-service-bucket/dir2/file1/



Create access key | IAM | Global
S3 buckets | S3 | ap-south-1
Download Zoom for Windows
ap-south-1.console.aws.amazon.com/s3/home?region=ap-south-1#
Microsoft Learn | Mi...
Learning | Future Re...
Gmail
Untitled document...
All Bookmarks
Mumbai ▾
yashwant ▾
EC2
AWS Auto Scaling
VPC
S3
Amazon S3
General purpose buckets
Directory buckets
Table buckets New
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
View Storage Lens dashboard
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Account snapshot - *updated every 24 hours* All AWS Regions

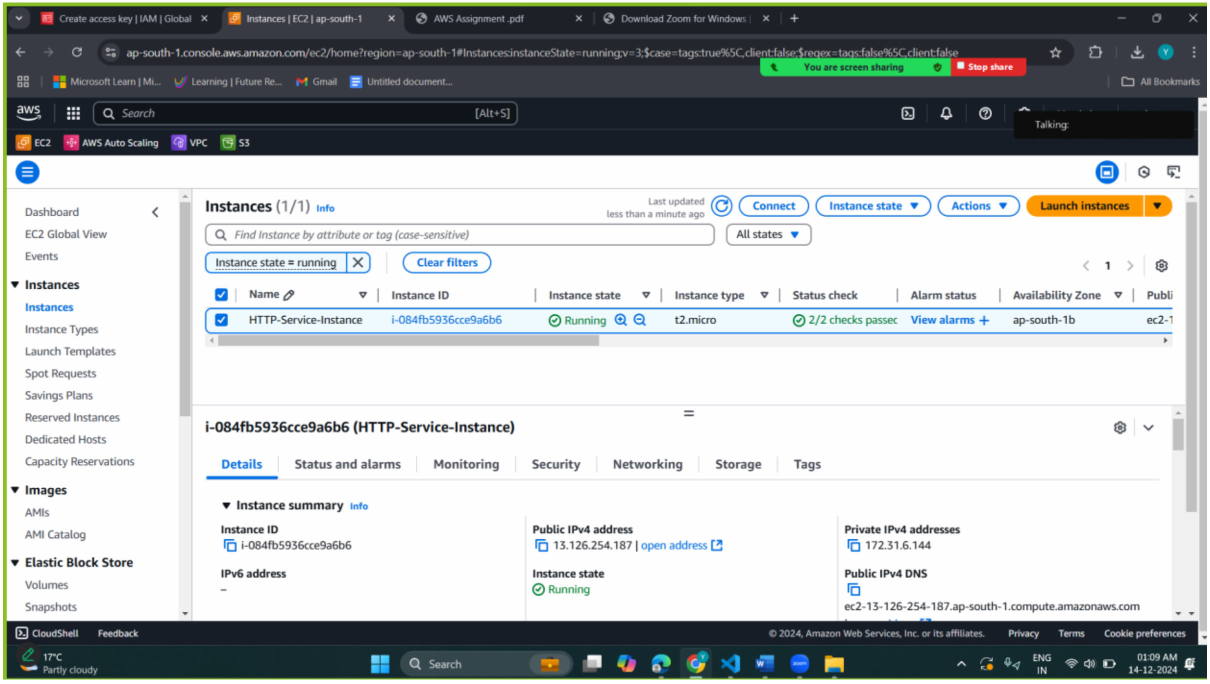
Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. Learn more

General purpose buckets | Directory buckets

General purpose buckets (3) Info All AWS Regions

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
○ e9-http-service-bucket	Asia Pacific (Mumbai) ap-south-1	View analyzer for ap-south-1	December 13, 2024, 23:54:17 (UTC+05:30)
○ my360a	US East (N. Virginia) us-east-1	View analyzer for us-east-1	October 14, 2024, 14:31:37 (UTC+05:30)
○ yashwant	US East (N. Virginia) us-east-1	View analyzer for us-east-1	October 14, 2024, 16:08:20 (UTC+05:30)



Instances (1/1) Info

☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Publi
☑	HTTP-Service-Instance	i-084fb5936cce9a6b6	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1b	ec2-1

i-084fb5936cce9a6b6 (HTTP-Service-Instance)

Instance summary Info

Instance ID
i-084fb5936cce9a6b6

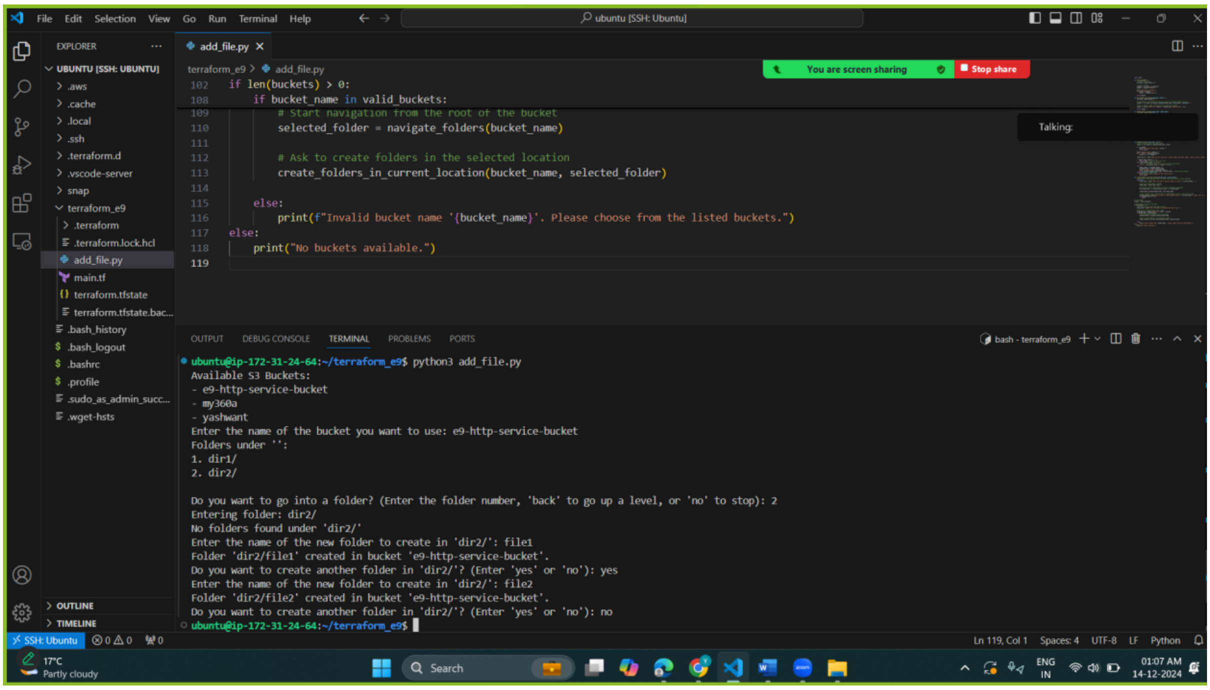
IPv6 address
–

Public IPv4 address
13.126.254.187 | open address

Instance state
Running

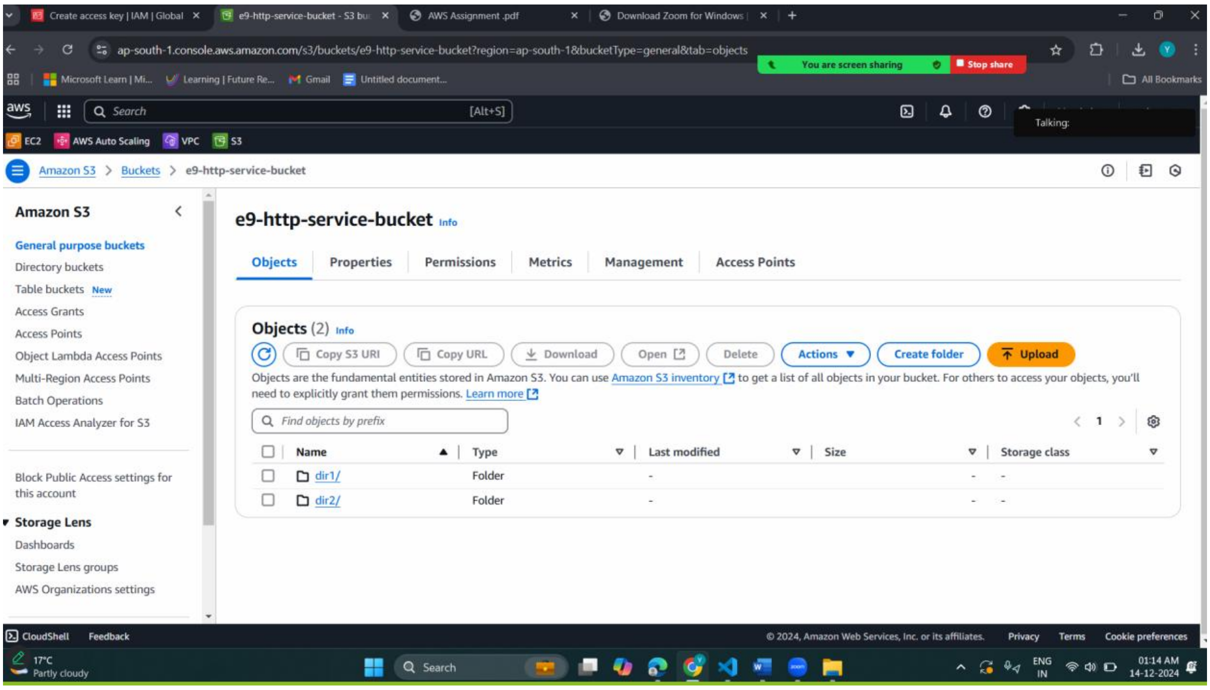
Private IPv4 addresses
172.31.6.144

Public IPv4 DNS
ec2-13-126-254-187.ap-south-1.compute.amazonaws.com



```python
if len(buckets) > 0:
    if bucket_name in valid_buckets:
        # Start navigation from the root of the bucket
        selected_folder = navigate_folders(bucket_name)

        # Ask to create folders in the selected location
        create_folders_in_current_location(bucket_name, selected_folder)

    else:
        print(f"Invalid bucket name '{bucket_name}'. Please choose from the listed buckets.")
else:
    print("No buckets available.")
```

```
ubuntu@ip-172-31-24-64:~/terraform_e9$ python3 add_file.py
Available S3 Buckets:
- e9-http-service-bucket
- my360a
- yashwant
Enter the name of the bucket you want to use: e9-http-service-bucket
Folders under '':
1. dir1/
2. dir2/

Do you want to go into a folder? (Enter the folder number, 'back' to go up a level, or 'no' to stop): 2
Entering folder: dir2/
No folders found under 'dir2/'
Enter the name of the new folder to create in 'dir2/': file1
Folder 'dir2/file1' created in bucket 'e9-http-service-bucket'.
Do you want to create another folder in 'dir2/'? (Enter 'yes' or 'no'): yes
Enter the name of the new folder to create in 'dir2/': file2
Folder 'dir2/file2' created in bucket 'e9-http-service-bucket'.
Do you want to create another folder in 'dir2/'? (Enter 'yes' or 'no'): no
ubuntu@ip-172-31-24-64:~/terraform_e9$
```

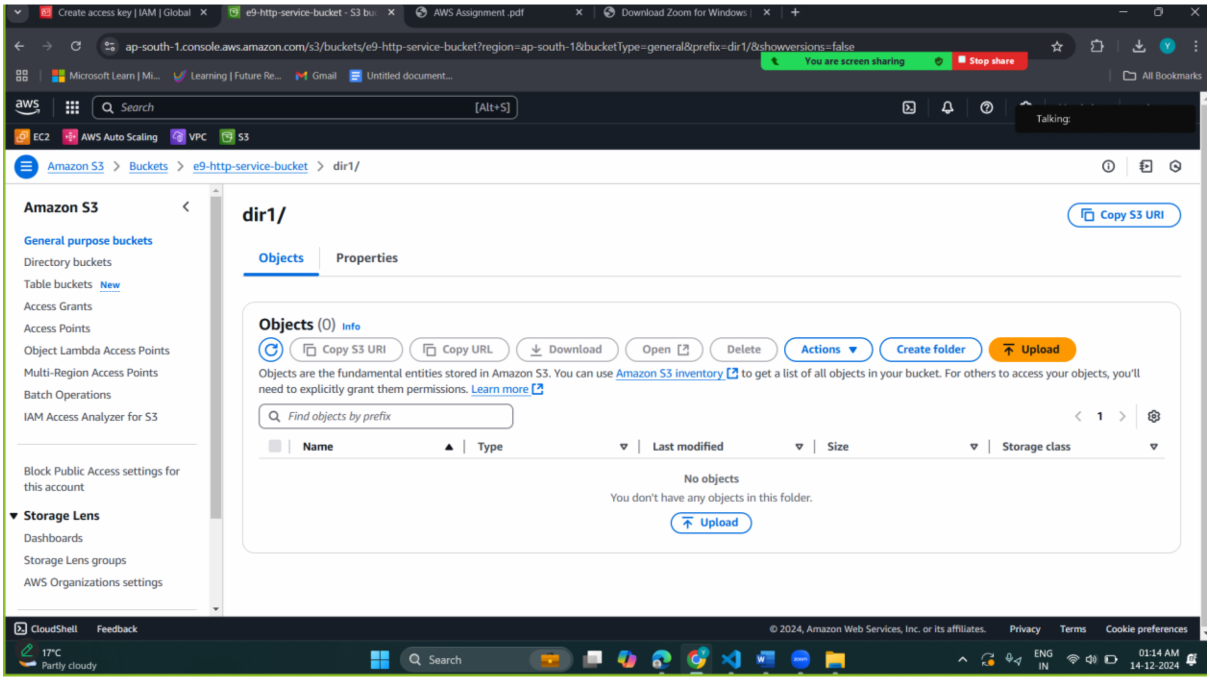


e9-http-service-bucket Info

Objects (2) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name	Type	Last modified	Size	Storage class
□	dir1/	Folder	-	-	-
□	dir2/	Folder	-	-	-



ap-south-1.console.aws.amazon.com/s3/buckets/e9-http-service-bucket?region=ap-south-1&bucketType=general&prefix=dir1/&showversions=false

Amazon S3 > Buckets > e9-http-service-bucket > dir1/

dir1/

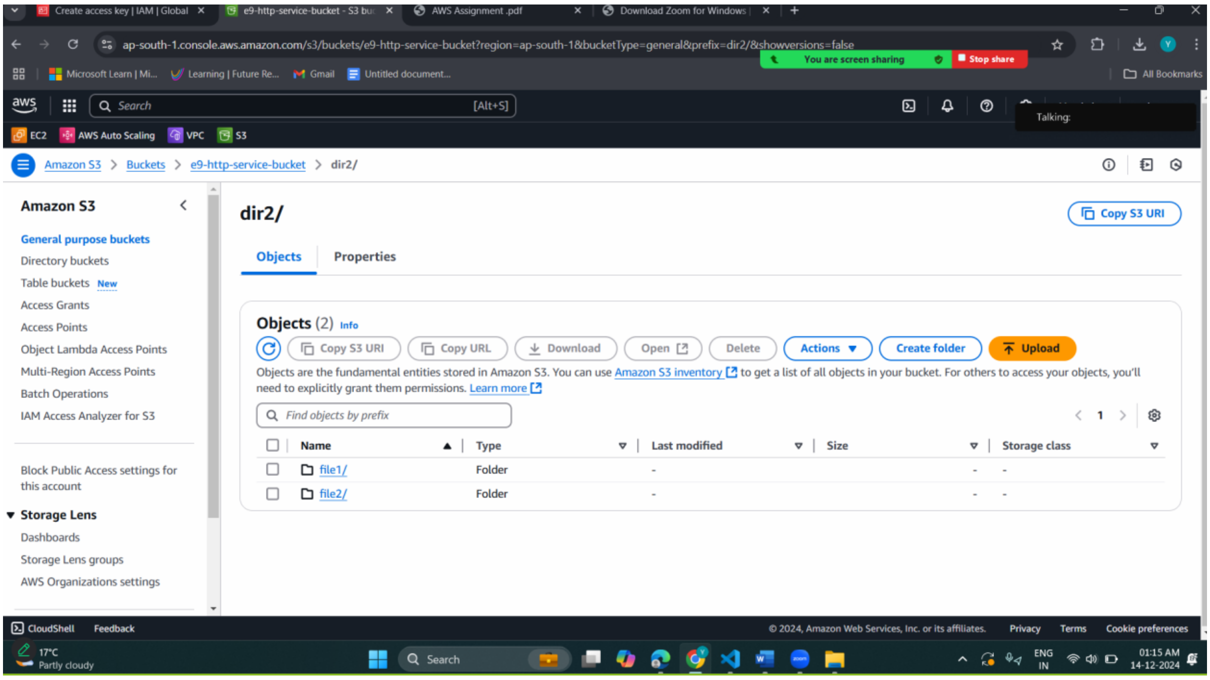
Objects (0) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name	Type	Last modified	Size	Storage class

No objects

You don't have any objects in this folder.



ap-south-1.console.aws.amazon.com/s3/buckets/e9-http-service-bucket?region=ap-south-1&bucketType=general&prefix=dir2/&showversions=false

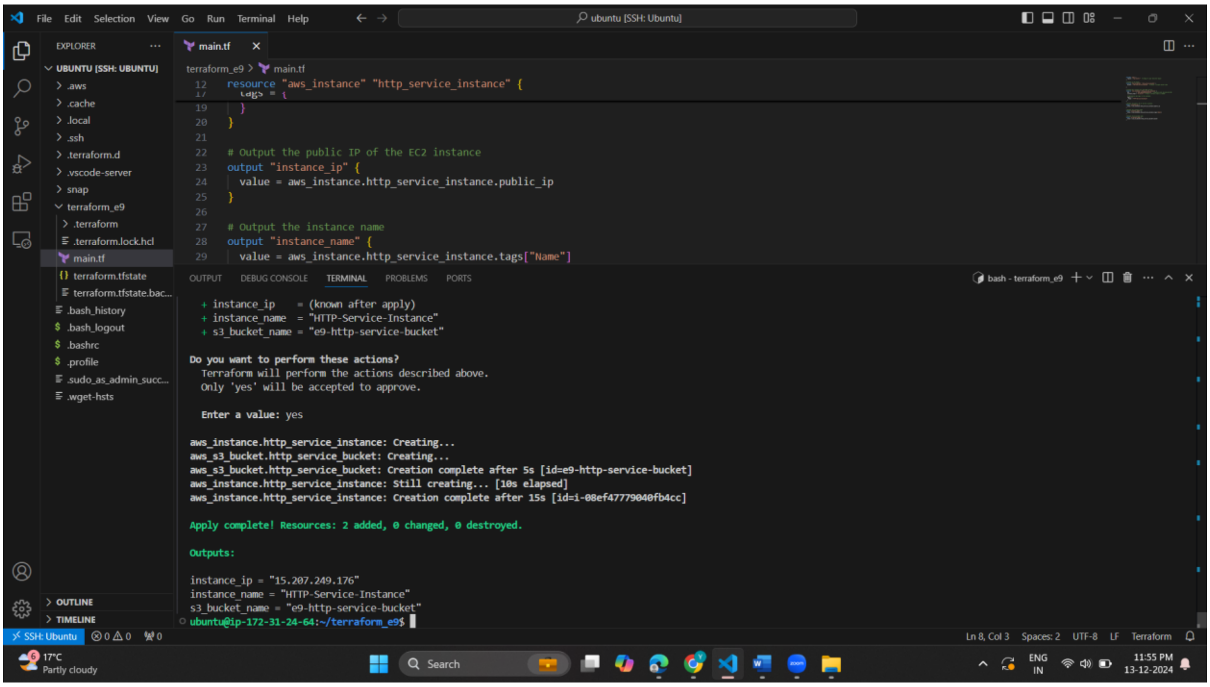
Amazon S3 > Buckets > e9-http-service-bucket > dir2/

dir2/

Objects (2) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name	Type	Last modified	Size	Storage class
□	file1/	Folder	-	-	-
□	file2/	Folder	-	-	-



```
resource "aws_instance" "http_service_instance" {
  tags = {
  }
}

# Output the public IP of the EC2 instance
output "instance_ip" {
  value = aws_instance.http_service_instance.public_ip
}

# Output the instance name
output "instance_name" {
  value = aws_instance.http_service_instance.tags["Name"]
```

```
  + instance_ip    = (known after apply)
  + instance_name  = "HTTP-Service-Instance"
  + s3_bucket_name = "e9-http-service-bucket"

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.http_service_instance: Creating...
aws_s3_bucket.http_service_bucket: Creating...
aws_s3_bucket.http_service_bucket: Creation complete after 5s [id=e9-http-service-bucket]
aws_instance.http_service_instance: Still creating... [10s elapsed]
aws_instance.http_service_instance: Creation complete after 15s [id=i-08ef47779040fb4cc]

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

Outputs:

instance_ip = "15.207.249.176"
instance_name = "HTTP-Service-Instance"
s3_bucket_name = "e9-http-service-bucket"
ubuntu@ip-172-31-24-64:~/terraform_e9$
```

Clean Up:

After the testing is complete, you can delete the AWS resources to avoid extra charges by running:

terraform destroy

1: Create a GitHub Repository

Log into GitHub:

![alt text]({A33AFECB-1D7F-49A4-963D-E847E5E5095F}.png)

Visit GitHub and log in with your GitHub account credentials.

Create a New Repository:

Repository Name: equip9

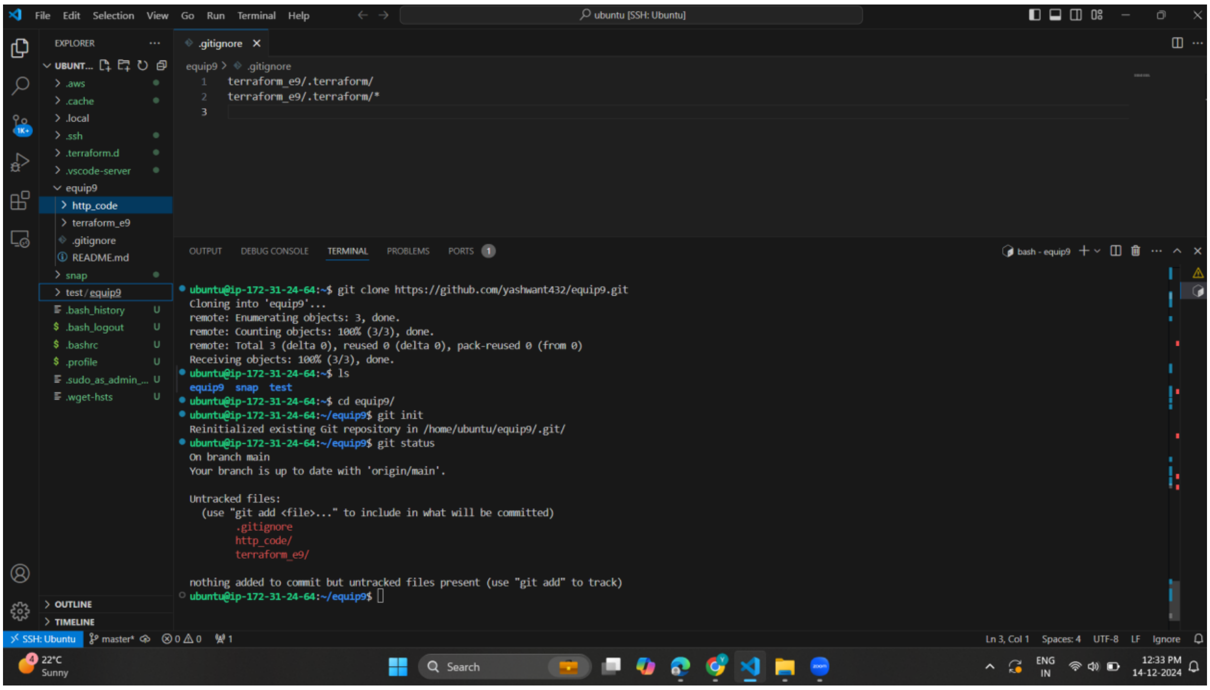
2: Initialize a Local Git Repository

Navigate to project directory: Open a terminal/command prompt and navigate to the root folder of your project (where your README.md and project files are located).

cd /home/ubuntu/equip9

Initialize a Git repository: Run the following command to initialize the local Git repository:

git init



```
terraform_e9/.terraform/
terraform_e9/.terraform/*
```

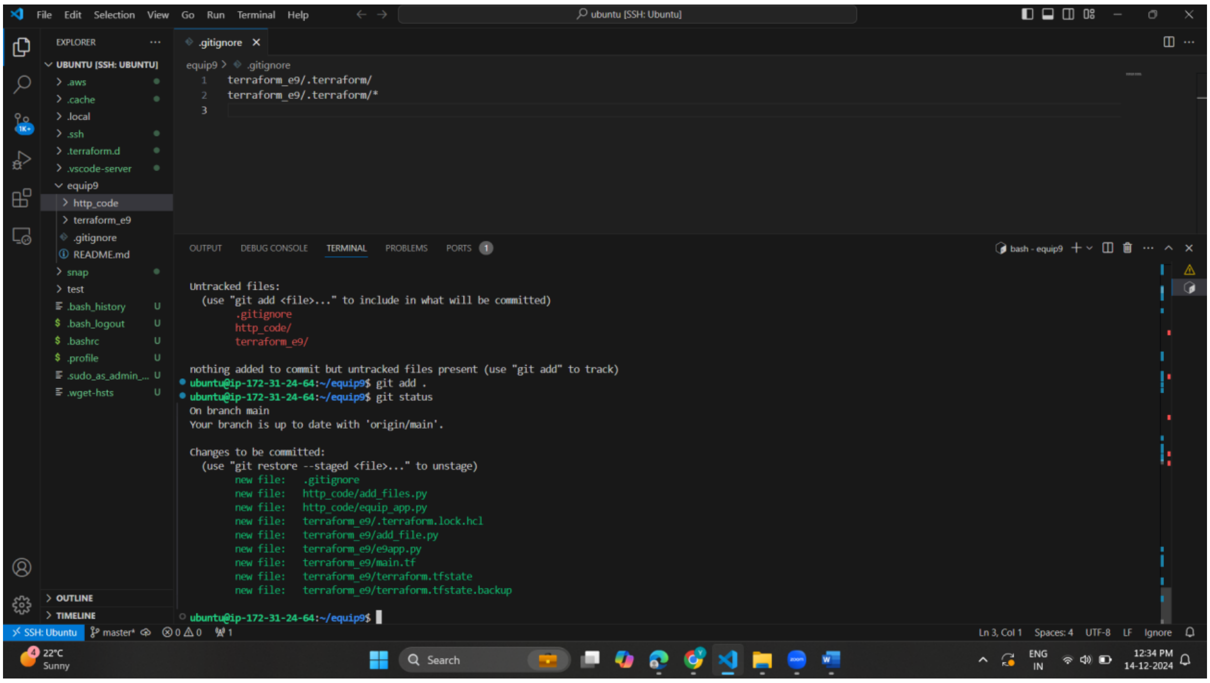
```
ubuntu@ip-172-31-24-64:~$ git clone https://github.com/yashwant432/equip9.git
Cloning into 'equip9'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
ubuntu@ip-172-31-24-64:~$ ls
equip9  snap  test
ubuntu@ip-172-31-24-64:~$ cd equip9/
ubuntu@ip-172-31-24-64:~/equip9$ git init
Reinitialized existing Git repository in /home/ubuntu/equip9/.git/
ubuntu@ip-172-31-24-64:~/equip9$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        http_code/
        terraform_e9/

nothing added to commit but untracked files present (use "git add" to track)
ubuntu@ip-172-31-24-64:~/equip9$ 
```

3: Add Files to the Repository

Add all files to the staging area: Run the following command to add all files in your project to the Git staging area:

git add .

screenshot ![alt text](image-19.png)



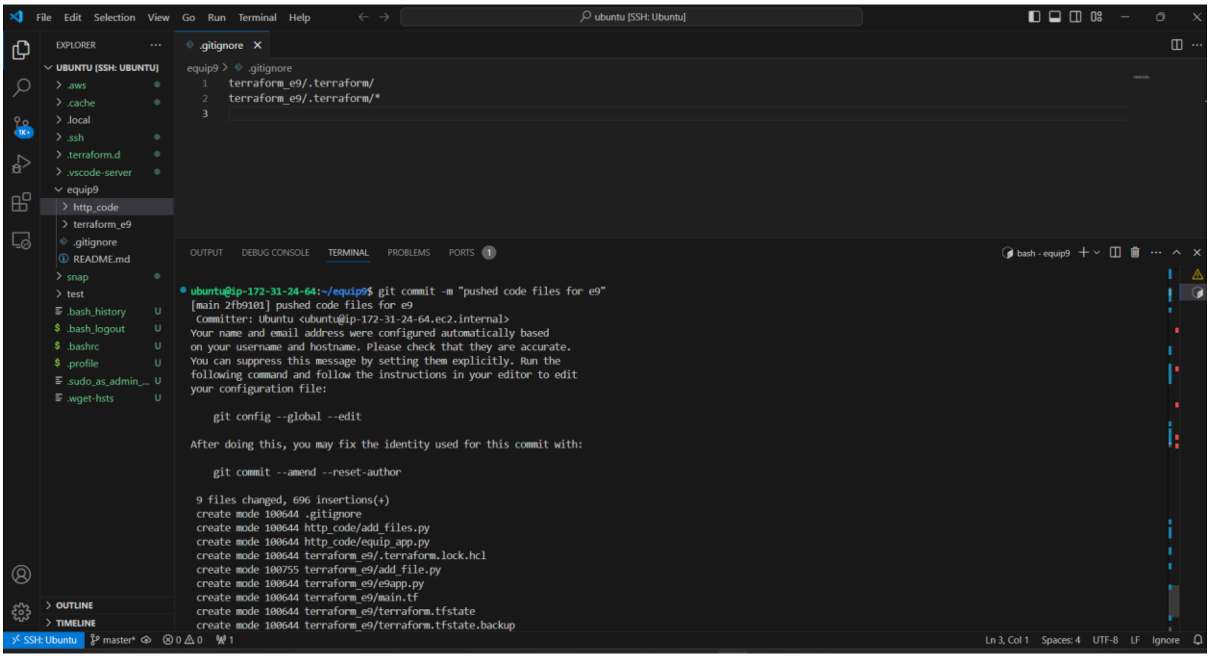
```
terraform_e9/.terraform/
terraform_e9/.terraform/*
```

```
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        http_code/
        terraform_e9/

nothing added to commit but untracked files present (use "git add" to track)
ubuntu@ip-172-31-24-64:~/equip9$ git add .
ubuntu@ip-172-31-24-64:~/equip9$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   .gitignore
        new file:   http_code/add_files.py
        new file:   http_code/equip_app.py
        new file:   terraform_e9/.terraform.lock.hcl
        new file:   terraform_e9/add_file.py
        new file:   terraform_e9/e9app.py
        new file:   terraform_e9/main.tf
        new file:   terraform_e9/terraform.tfstate
        new file:   terraform_e9/terraform.tfstate.backup

ubuntu@ip-172-31-24-64:~/equip9$ 
```

Commit the changes: Commit changes with a descriptive message:

git commit -m "Initial commit with HTTP service and Terraform deployment"

screenshot ![alt text](image-20.png)



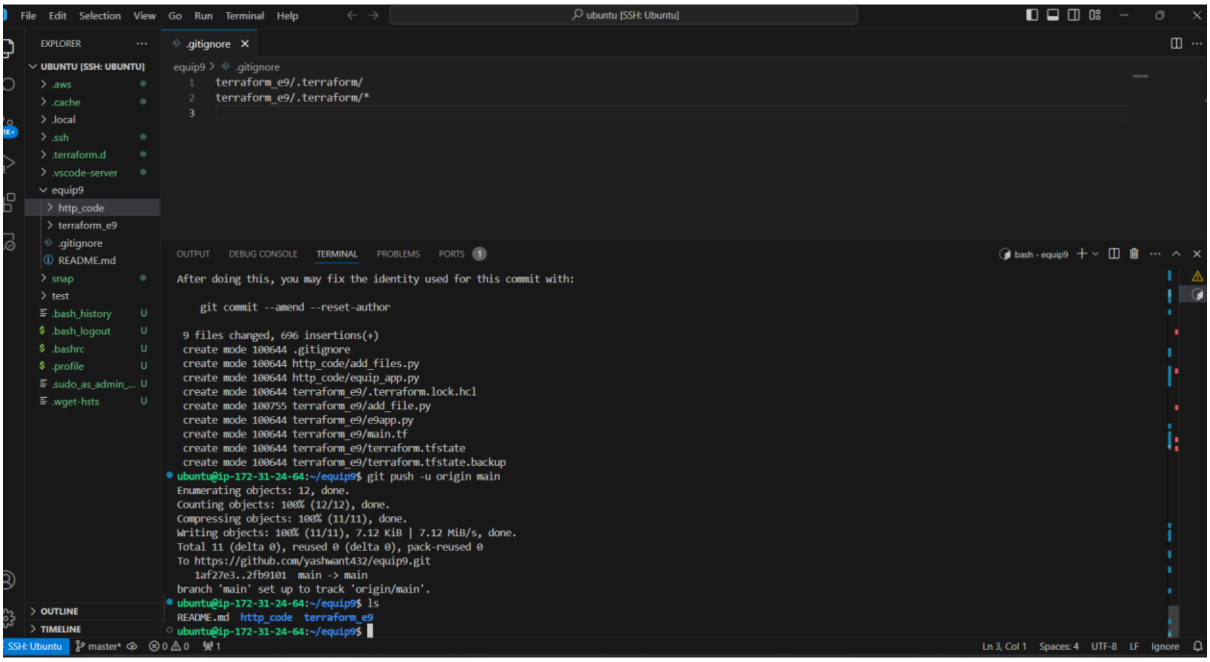
```
terraform_e9/.terraform/
terraform_e9/.terraform/*
```

```
ubuntu@ip-172-31-24-64:~/equip9$ git commit -m "pushed code files for e9"
[main 2fb9101] pushed code files for e9
 Committer: Ubuntu <ubuntu@ip-172-31-24-64.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 9 files changed, 696 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 http_code/add_files.py
 create mode 100644 http_code/equip_app.py
 create mode 100644 terraform_e9/.terraform.lock.hcl
 create mode 100755 terraform_e9/add_file.py
 create mode 100644 terraform_e9/e9app.py
 create mode 100644 terraform_e9/main.tf
 create mode 100644 terraform_e9/terraform.tfstate
 create mode 100644 terraform_e9/terraform.tfstate.backup
```

5: Push Code to GitHub

Push the code to the repository: Push code to the main branch on GitHub using:

git push -u origin main



```
terraform_e9/.terraform/
terraform_e9/.terraform/*
```

```
After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 9 files changed, 696 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 http_code/add_files.py
 create mode 100644 http_code/equip_app.py
 create mode 100644 terraform_e9/.terraform.lock.hcl
 create mode 100755 terraform_e9/add_file.py
 create mode 100644 terraform_e9/e9app.py
 create mode 100644 terraform_e9/main.tf
 create mode 100644 terraform_e9/terraform.tfstate
 create mode 100644 terraform_e9/terraform.tfstate.backup
ubuntu@ip-172-31-24-64:~/equip9$ git push -u origin main
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 7.12 KiB | 7.12 MiB/s, done.
Total 11 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/yashwant432/equip9.git
   1af27e3..2fb9101  main -> main
branch 'main' set up to track 'origin/main'.
ubuntu@ip-172-31-24-64:~/equip9$ ls
README.md  http_code  terraform_e9
ubuntu@ip-172-31-24-64:~/equip9$
```

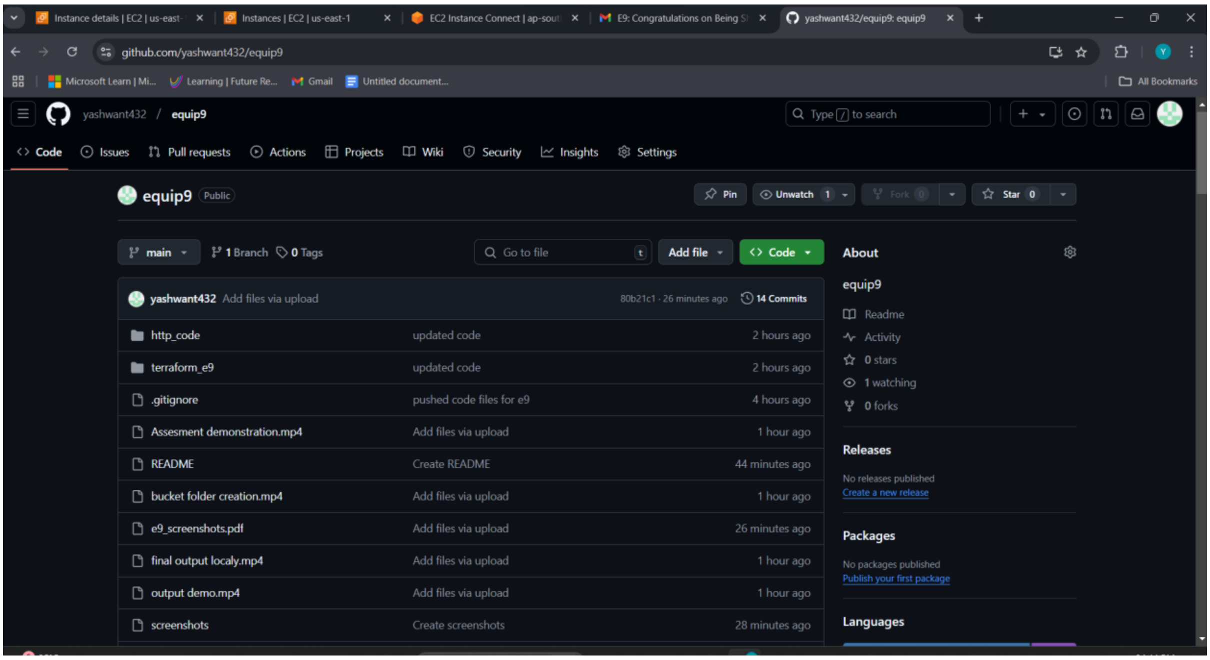
screenshot

Step 6: Verify on GitHub

Go to your GitHub repository: After pushing, visit your repository page on GitHub

https://github.com/yashwant432/equip9.git

screenshot ![alt text]({FA2DF79E-570D-45F0-9D7D-DDE8DEE0CC84}.png)



github.com/yashwant432/equip9

yashwant432 Add files via upload 80b21c1 · 26 minutes ago 14 Commits

http_code	updated code	2 hours ago
terraform_e9	updated code	2 hours ago
.gitignore	pushed code files for e9	4 hours ago
Assesment demonstration.mp4	Add files via upload	1 hour ago
README	Create README	44 minutes ago
bucket folder creation.mp4	Add files via upload	1 hour ago
e9_screenshots.pdf	Add files via upload	26 minutes ago
final output localy.mp4	Add files via upload	1 hour ago
output demo.mp4	Add files via upload	1 hour ago
screenshots	Create screenshots	28 minutes ago

About

equip9

Readme

Activity

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages