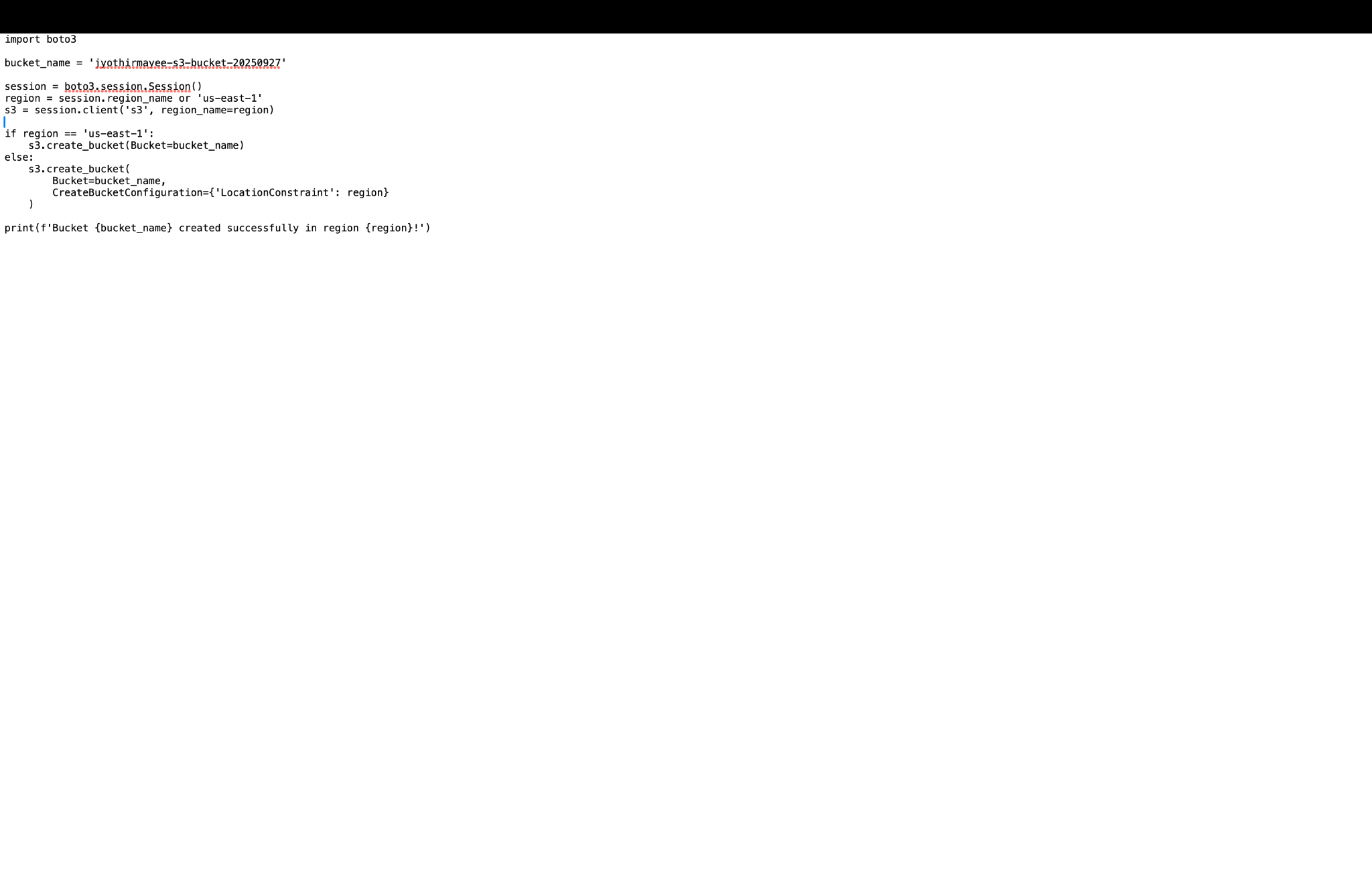
# Lab 3C - Amazon S3 with Boto3 (Python SDK)

## Step 1: Create an S3 Bucket:

In this step, a new S3 bucket is created using the Boto3 library. The Python script defines the bucket name, region, and logic to create the bucket. If the region is 'us-east-1', the bucket is created without location constraints. Otherwise, a location constraint parameter is passed.



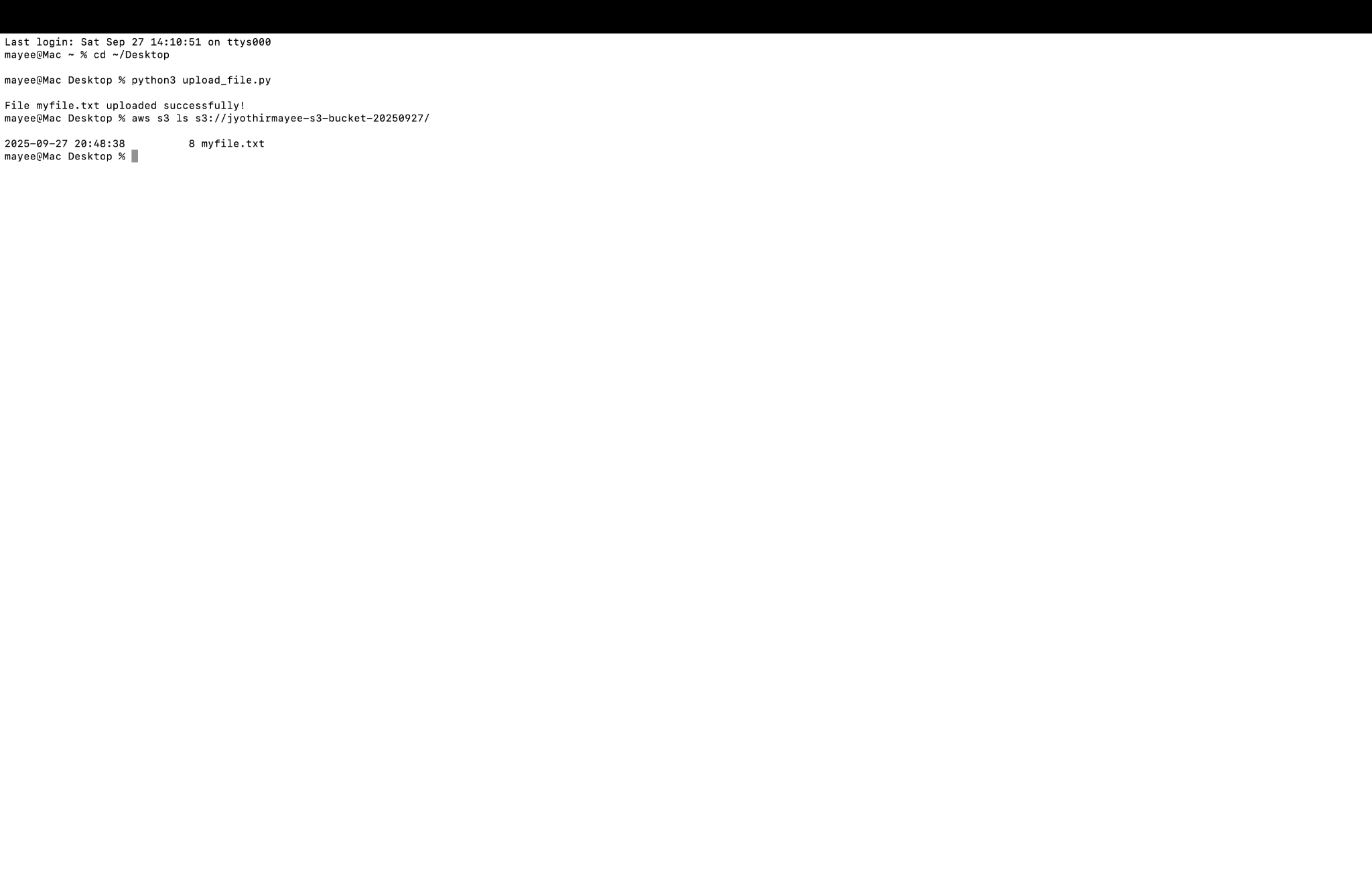
## Step 2: Upload a File to S3

Next, a text file is created and uploaded to the S3 bucket using the upload\_file method in Boto3. This confirms that the bucket is working as expected and able to store objects.



## Step 3: Verify Upload using AWS CLI

The AWS CLI is used to verify that the file was uploaded successfully. By listing the bucket contents with the aws s3 ls command, the uploaded file is displayed.



## Step 4: Download the File

In this step, the file is downloaded from the S3 bucket using the download\_file method in Boto3. This ensures that the file stored in the cloud can be retrieved locally.



## Step 5: Delete the File

After verifying the upload and download functionality, the uploaded file is deleted from the S3 bucket using the delete\_object method in Boto3.



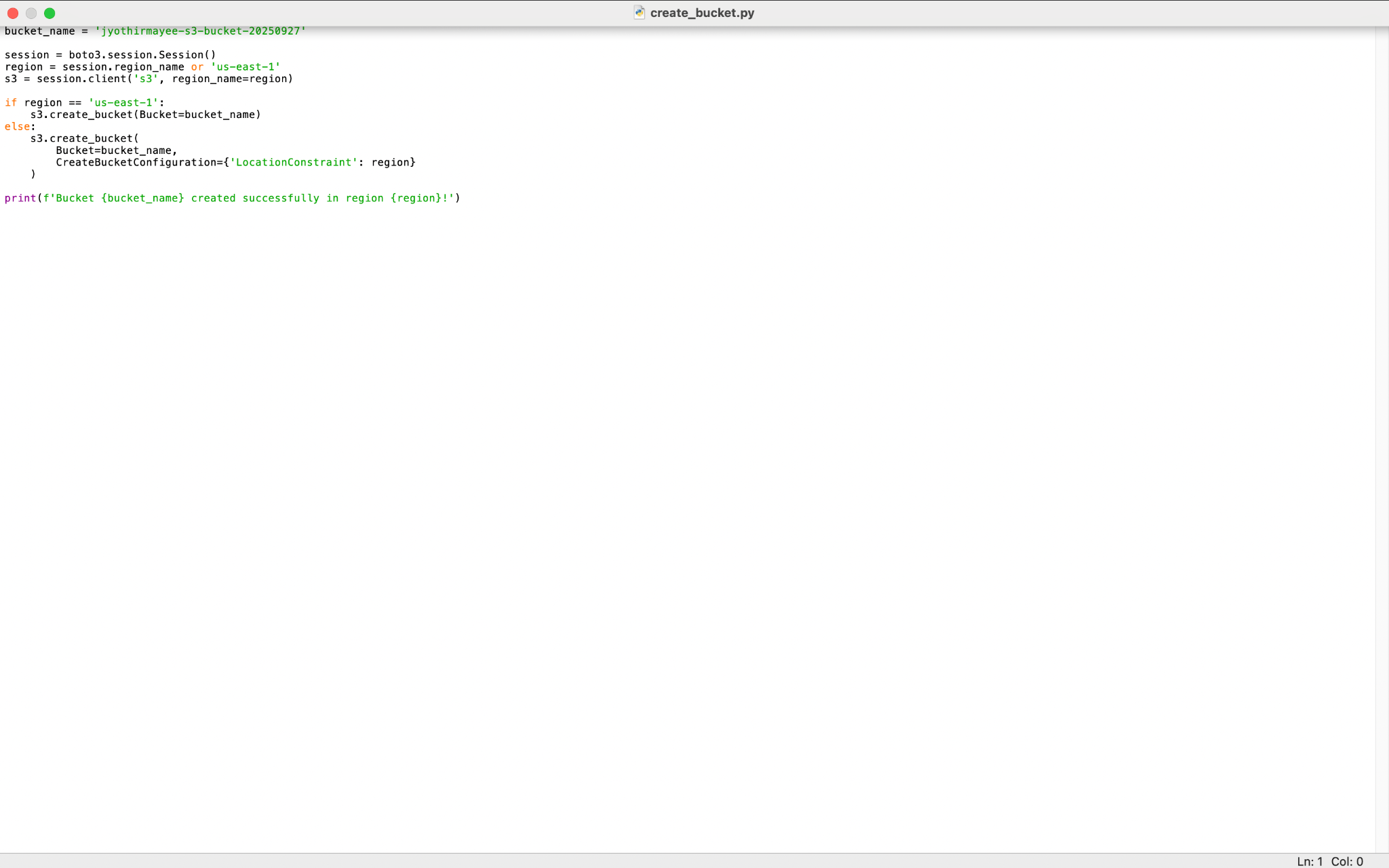
## Step 6: Enable Versioning

Finally, versioning is enabled on the S3 bucket using the put\_bucket\_versioning method. This feature allows multiple versions of the same file to be stored in the bucket, helping with recovery in case of accidental overwrites or deletions.



## Step 7: Validate Versioning with AWS CLI

The AWS CLI is again used to confirm that versioning is enabled on the bucket. This completes the lab demonstration.



## Step 8: Delete the Bucket

As a cleanup step, the S3 bucket is deleted using a Python script and verified with the AWS CLI. This ensures that no unnecessary resources are left running.

