**Tasks on S3 Bucket**

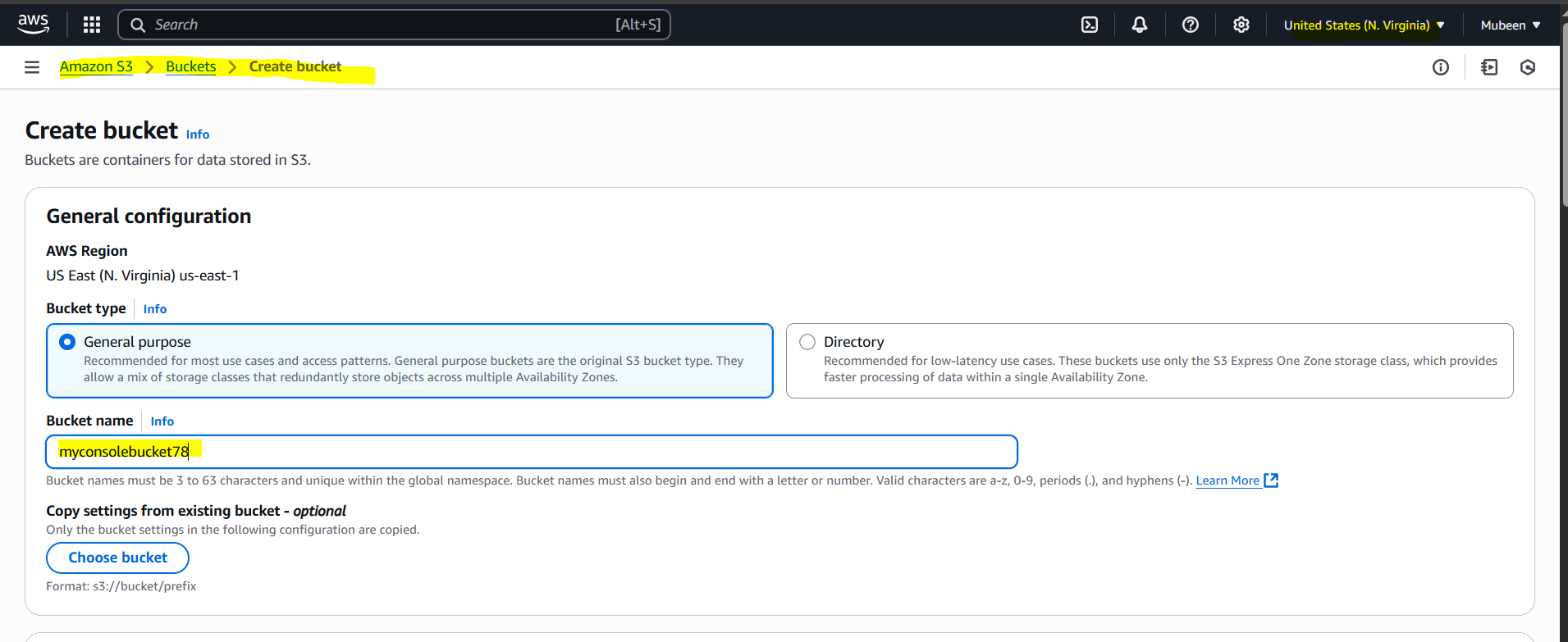
**1) Create s3 bucket and upload some objects to s3.   
  
Step 1: Create an S3 Bucket**

1. Log in to the AWS Management Console.

2. Navigate to the S3 dashboard.

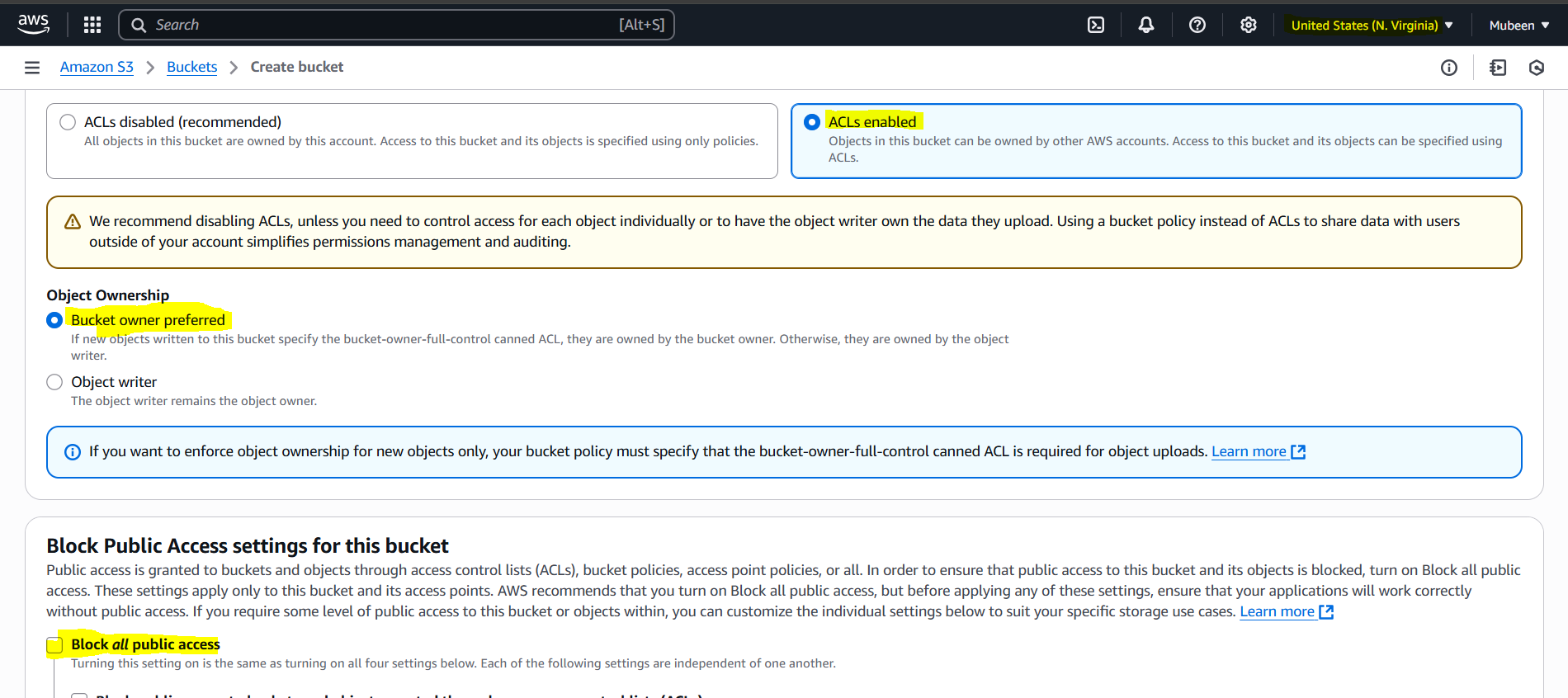
3. Click "Create bucket".

4. Enter a unique bucket name.

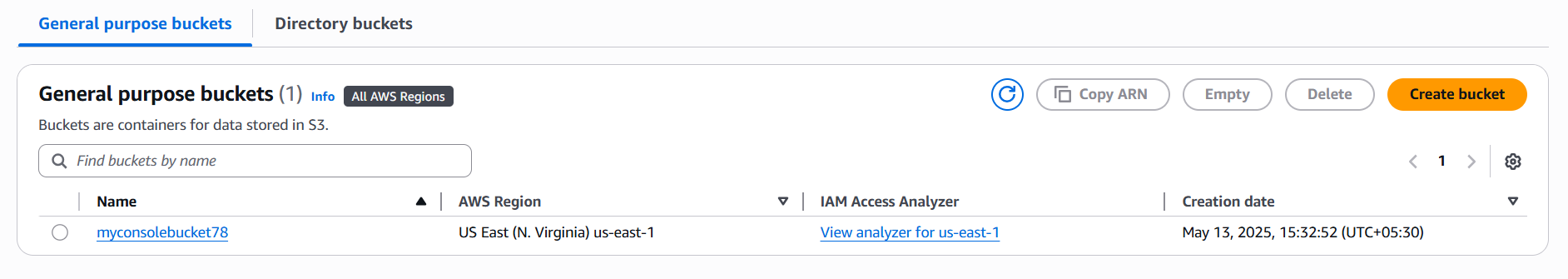


5. Choose a region.

6. Click "Create bucket".



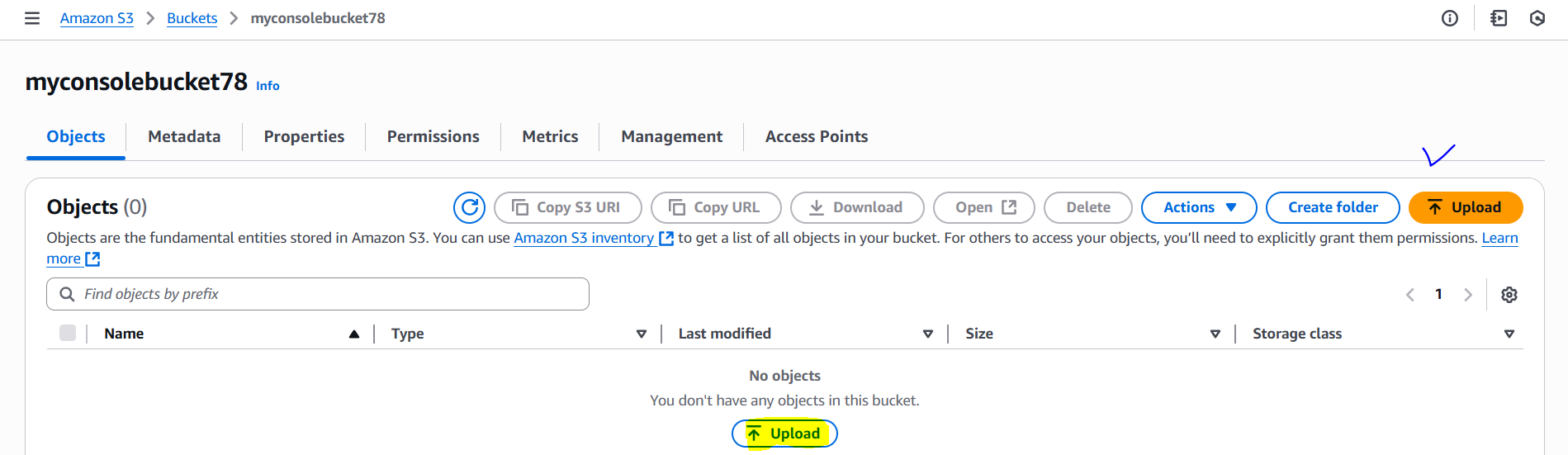
Full access for public is allowed since the creating of the bucket.



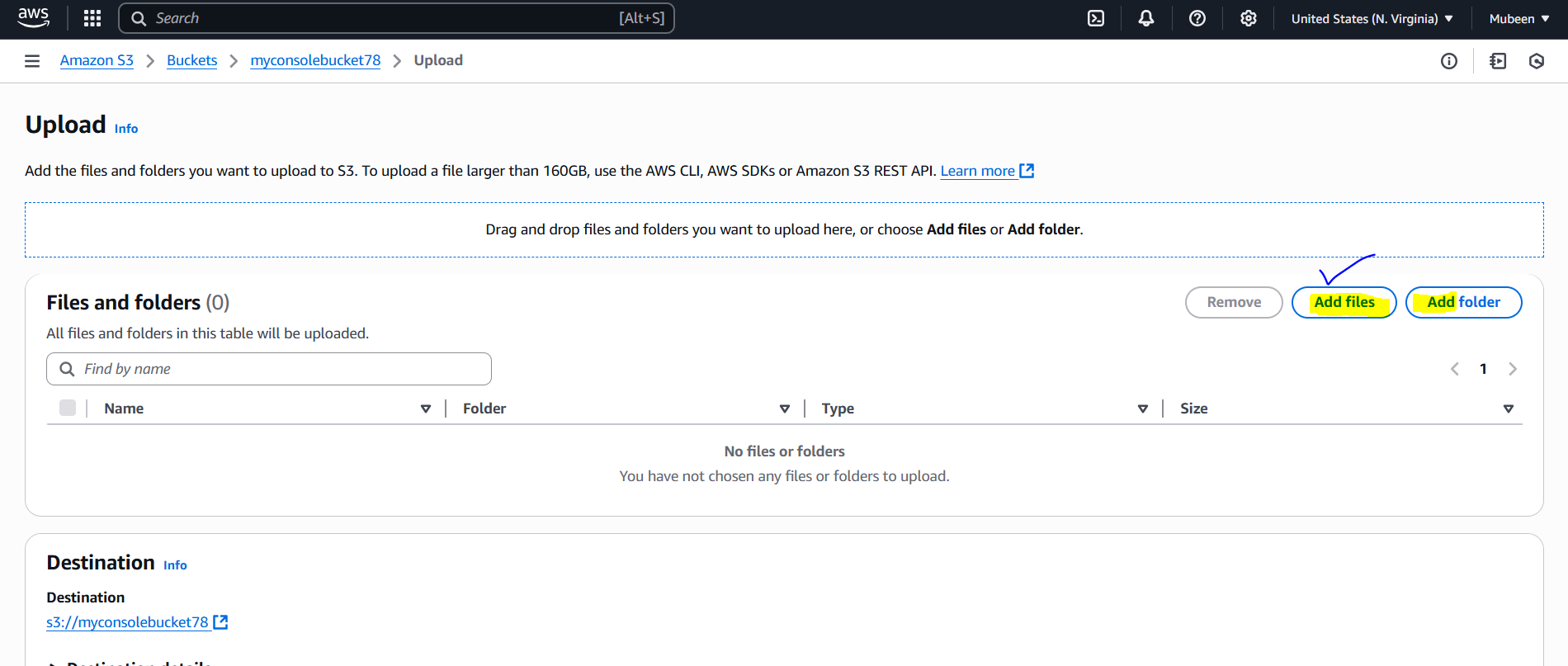
**Step 2: Upload Objects to S3**

1. Navigate to your newly created S3 bucket.

2. Click "Upload".

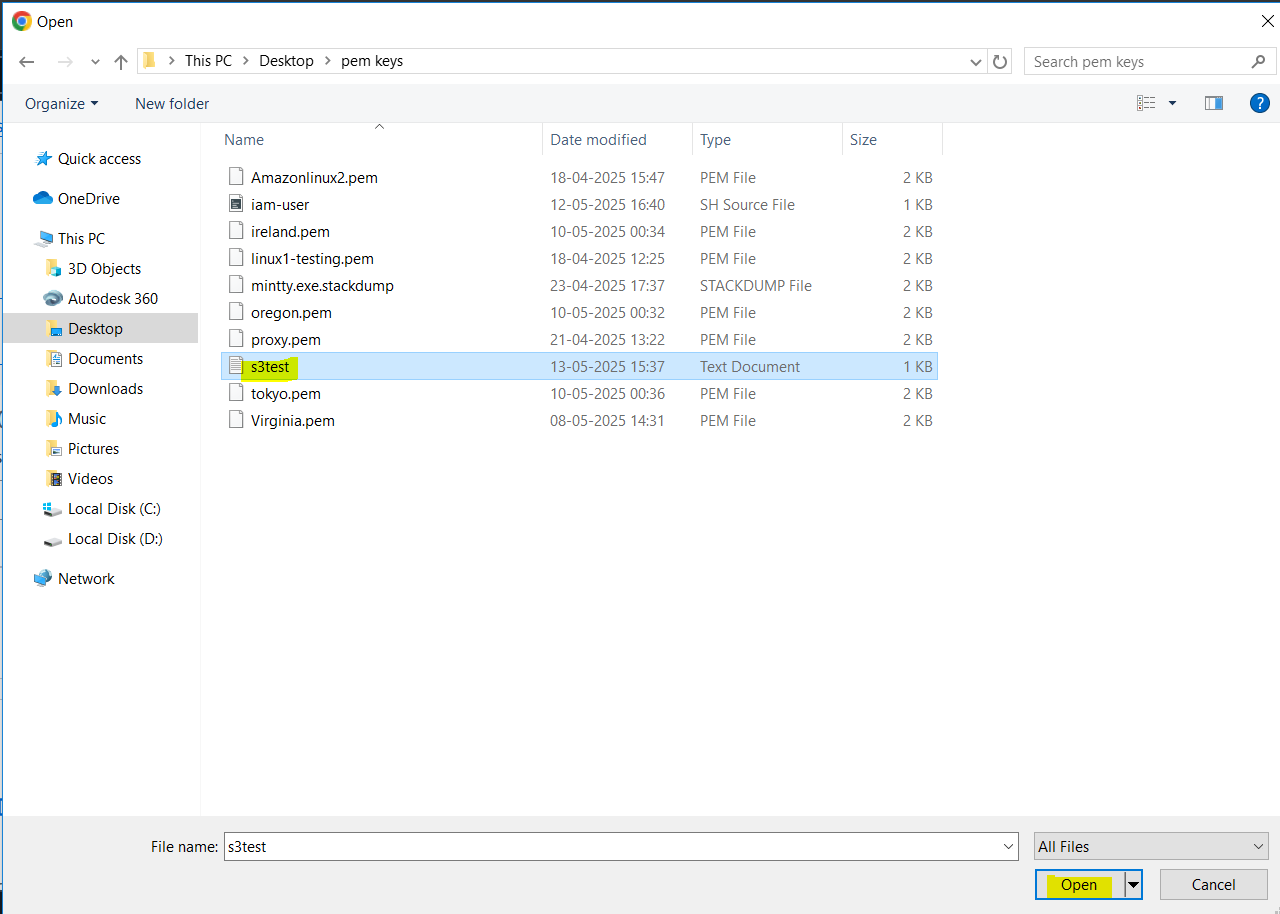


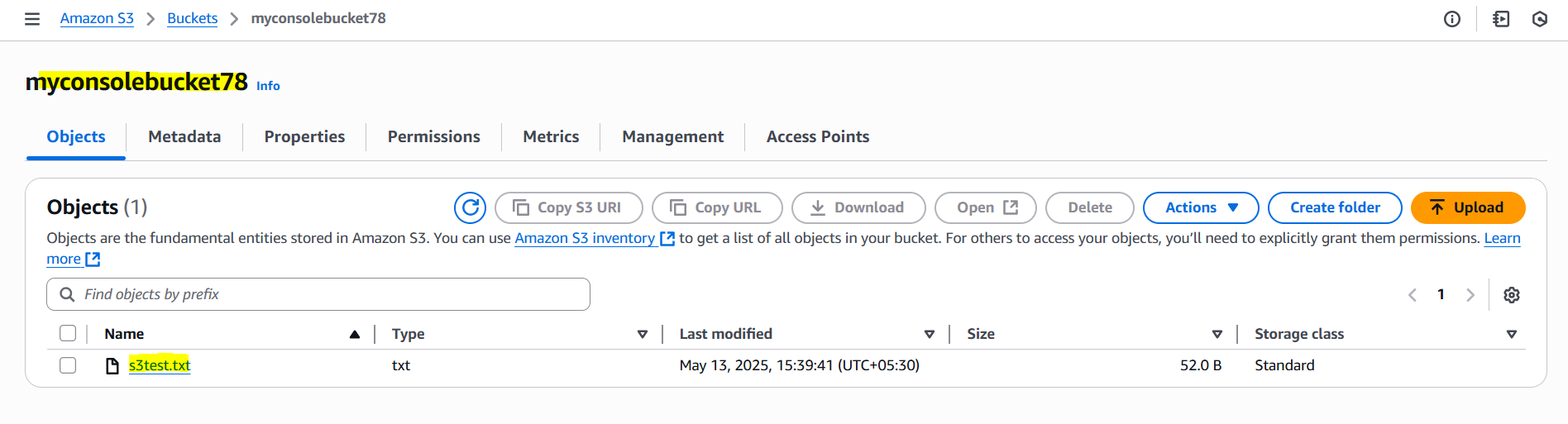
3. Click "Add files" or "Add folder".



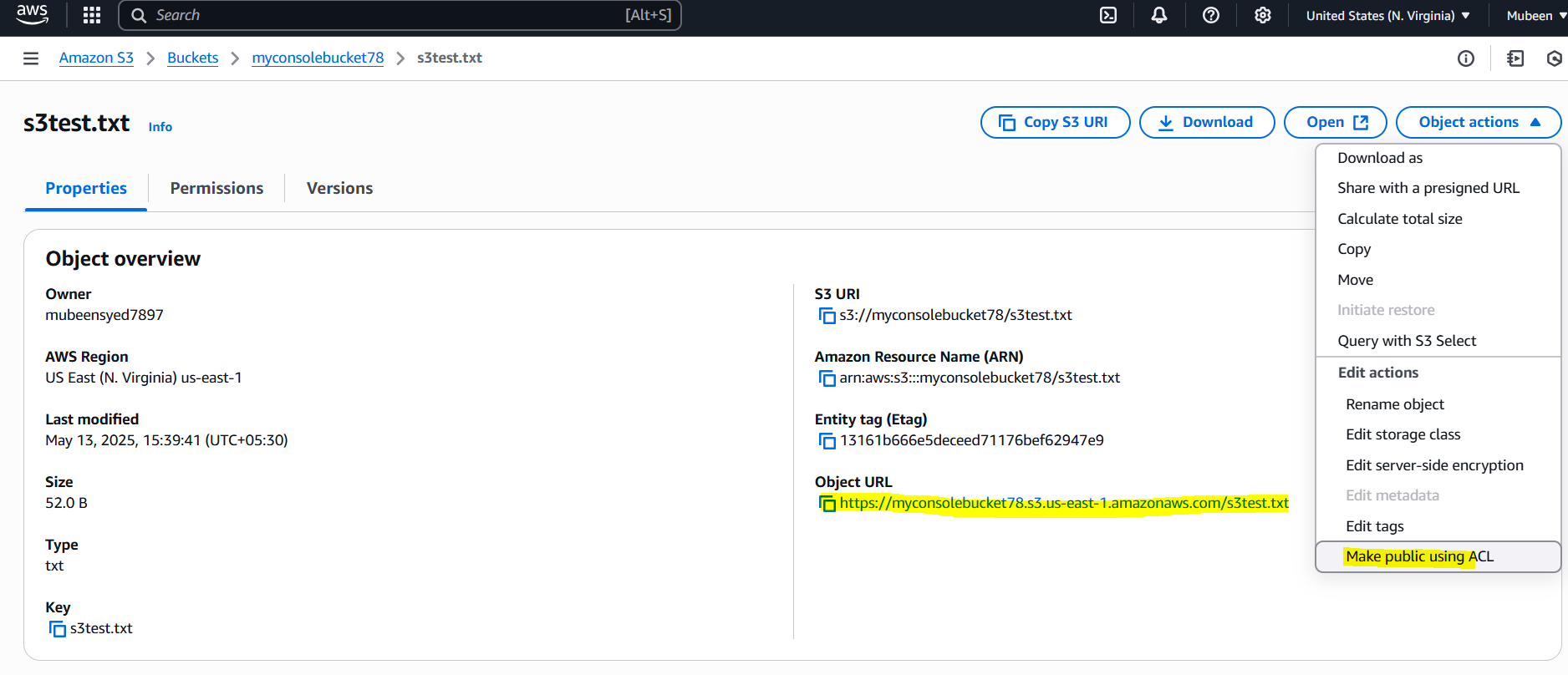
4. Select the files or folder you want to upload.

5. Click "Upload".

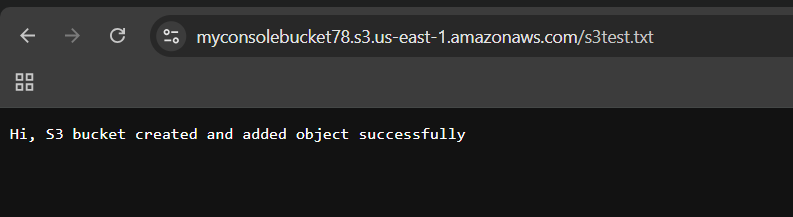




Add Public permissions to view the object through URL, by Object – Object actions – Make public using ACL.



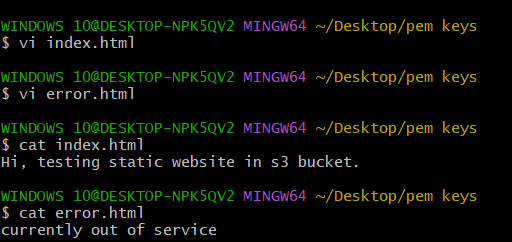
Output of URL.



**2) Deploy static website in s3 bucket.   
  
Step 1: Prepare Your Website Files**

1. Gather your static website files (HTML, CSS, JavaScript, images, etc.).

2. Ensure your website has an index.html file (or another specified index document).



**Step 2: Upload Your Website Files to S3 bucket myconsolebucket78**

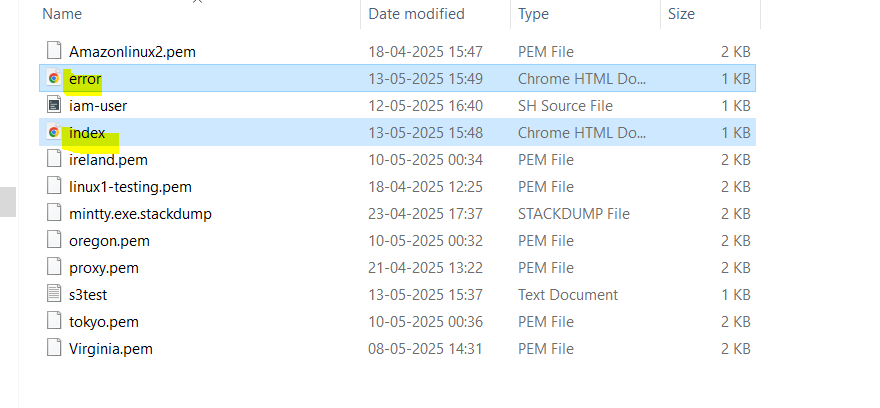
1. Navigate to your S3 bucket.

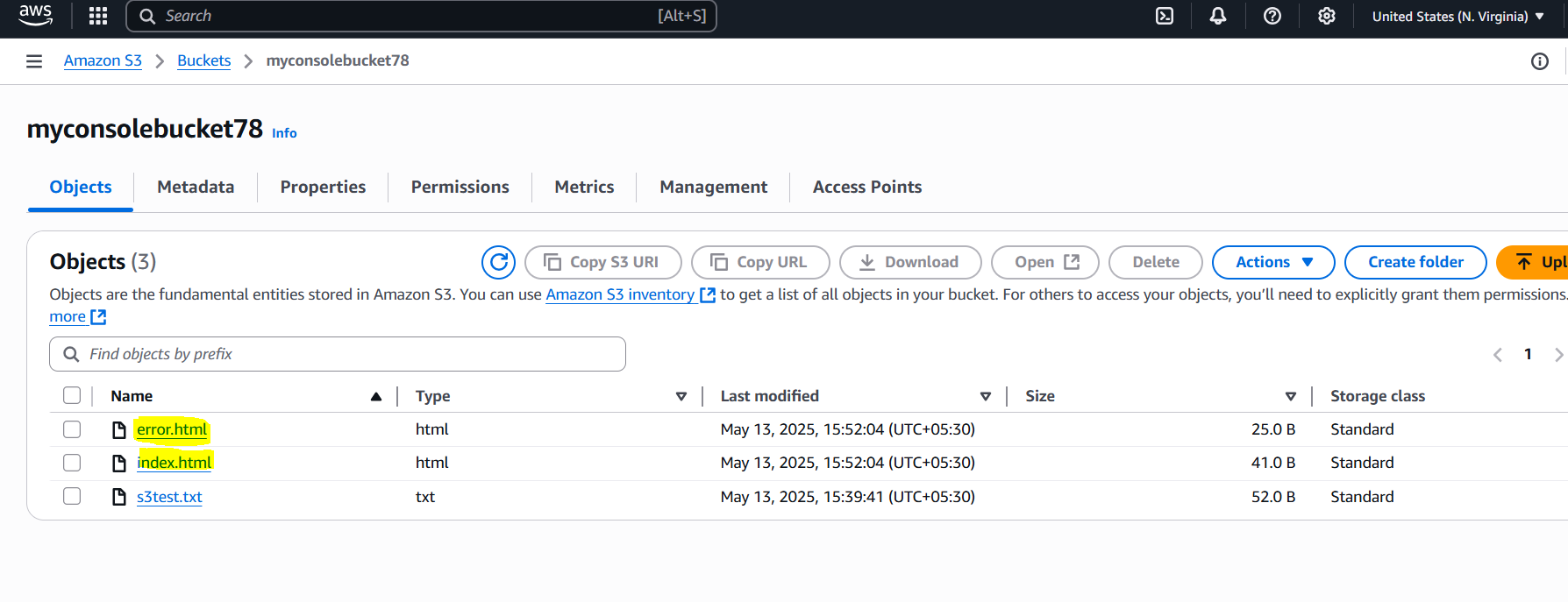
2. Click "Upload".

3. Select your website files.

4. Choose the storage class and other settings as needed.

5. Click "Upload".





**Step 3: Configure Bucket Policy for Public Access**

1. Navigate to your S3 bucket's Permissions tab.

2. Click "Bucket policy".

3. Add a policy to allow public access:

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "PublicReadGetObject",

"Effect": "Allow",

"Principal": "\*",

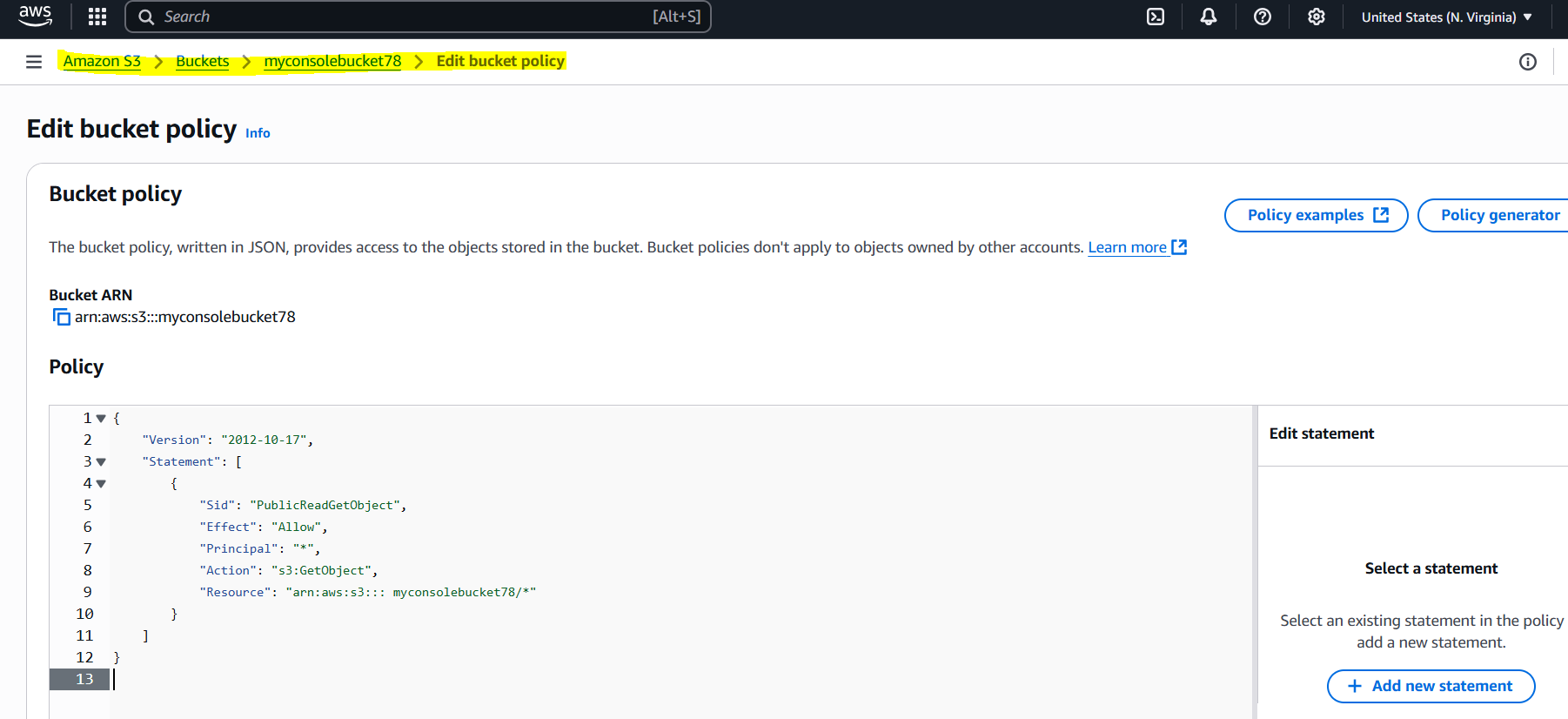
"Action": "s3:GetObject",

"Resource": "arn:aws:s3:::myconsolebucket78/\*"

}

]

}

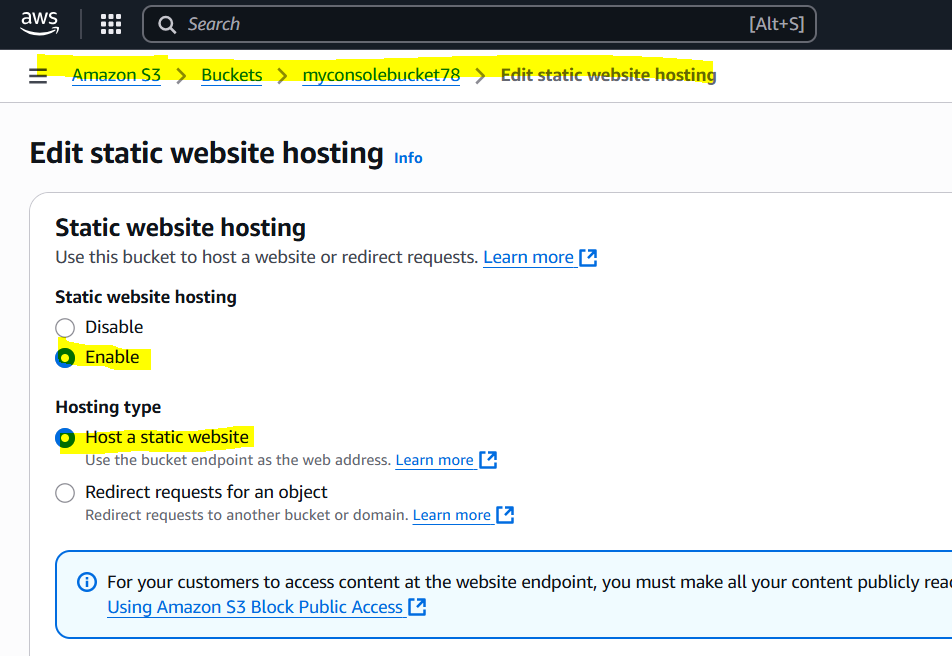


**Step 4: Enable Static Website Hosting**

1. Navigate to your S3 bucket's Properties tab.

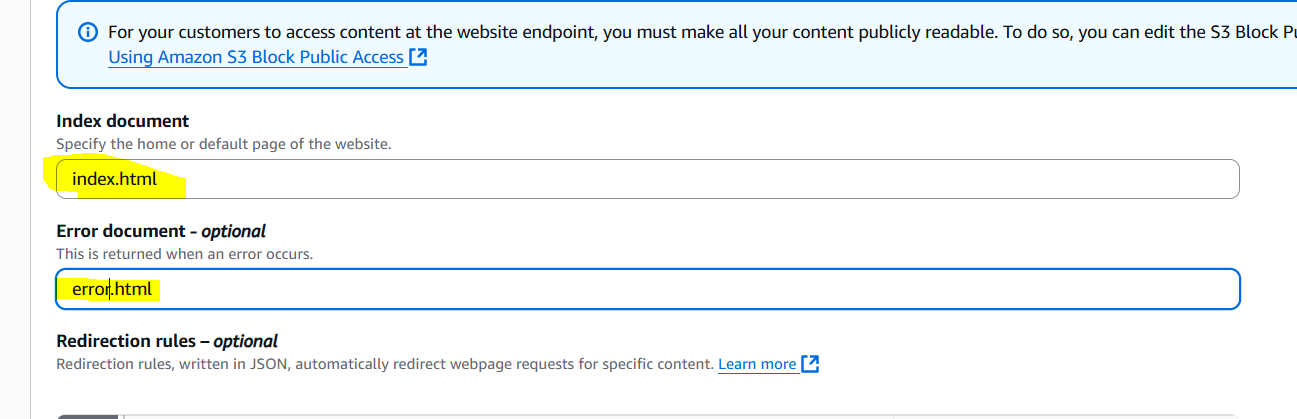
2. Click "Static website hosting".

3. Select "Enable".

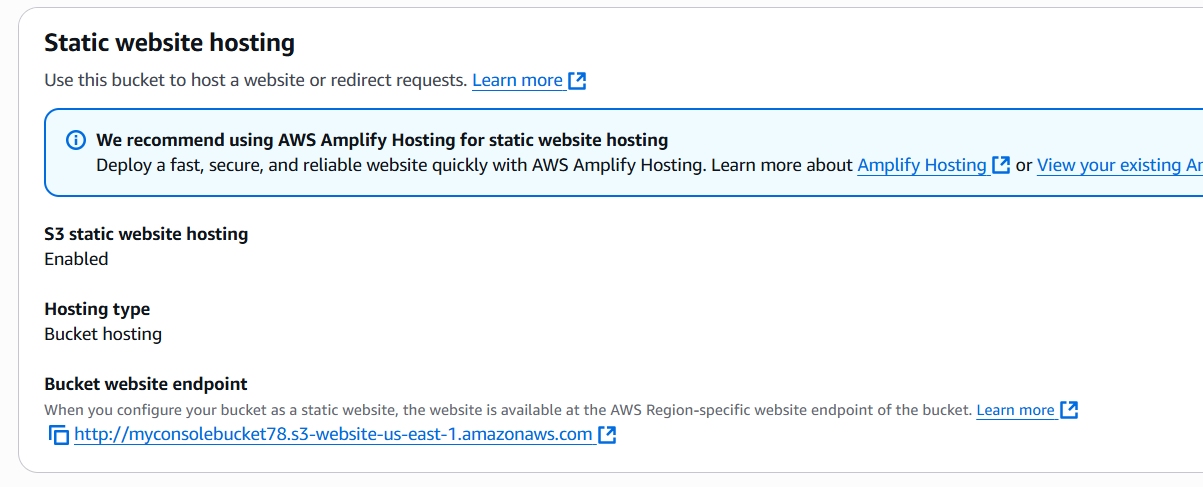


4. Enter the index document name (e.g., index.html).

5. Optionally, specify an error document.



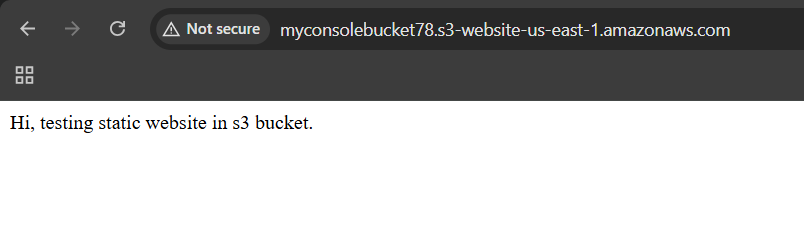
6. Click "Save changes".



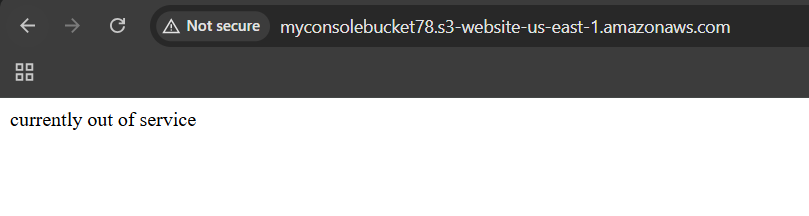
**Step 5: Access Your Website**

1. Note the endpoint URL provided in the Static website hosting section.

2. Access your website using the endpoint URL



Deleting index.html to check error.html.



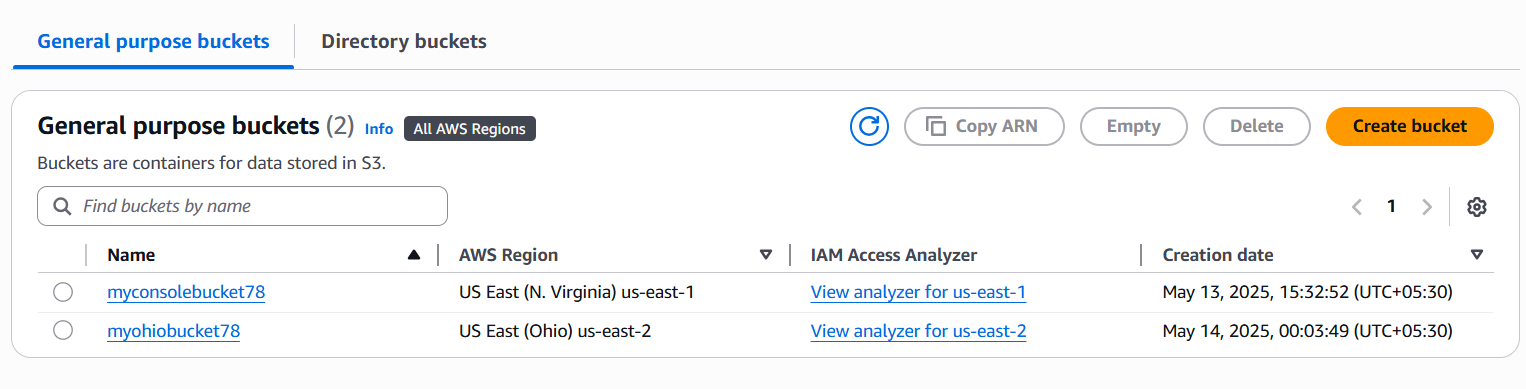
**3) Enable cross region replication on s3 buckets.**  
**Step 1: Enable Versioning on Both Buckets**

1. Go to the S3 dashboard and select the source bucket.

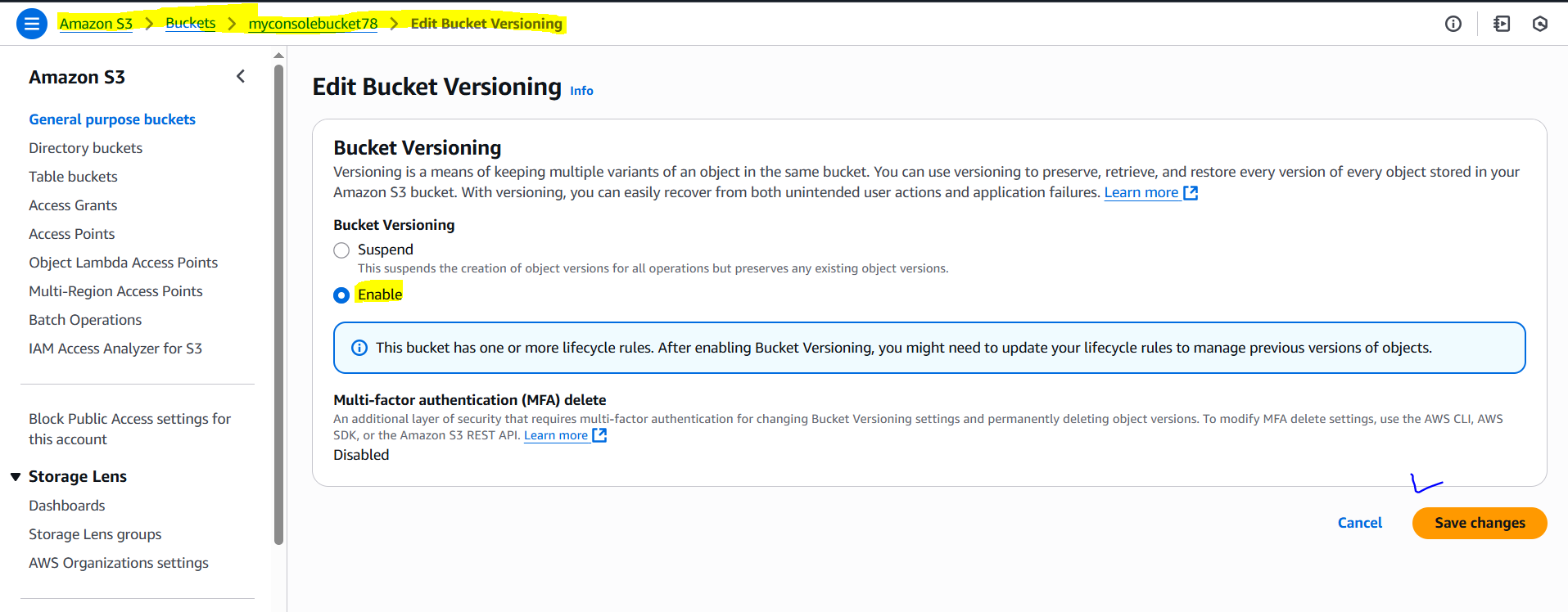
2. Click on "Properties" and then click on "Versioning".

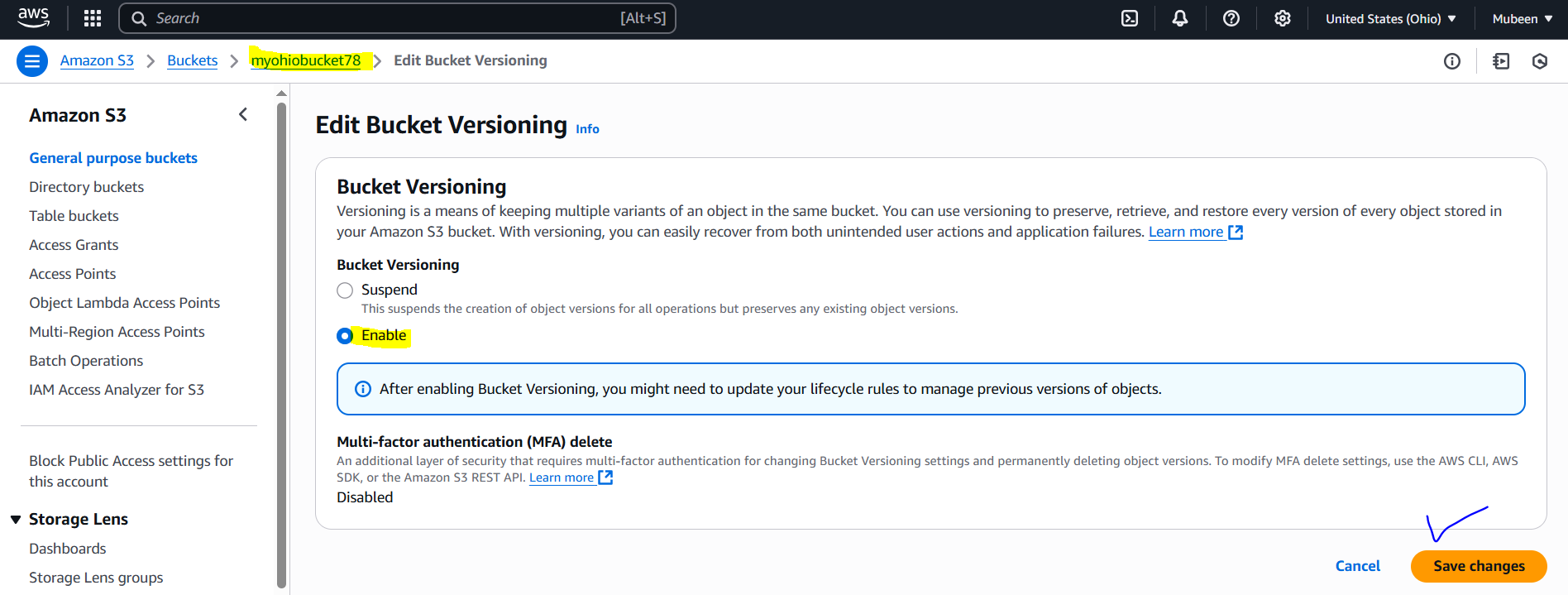
3. Click on "Edit" and select "Enable versioning".

4. Repeat the same steps for the destination bucket.



Taking **myconsolebucket78 (virginia)** as a **source bucket** and **myohiobucket78 (ohio)** as a **destination bucket**.





**Step 2: Enable Cross-Region Replication**

1. Go to the S3 dashboard and select the source bucket.

2. Click on "Properties" and then click on "Management".

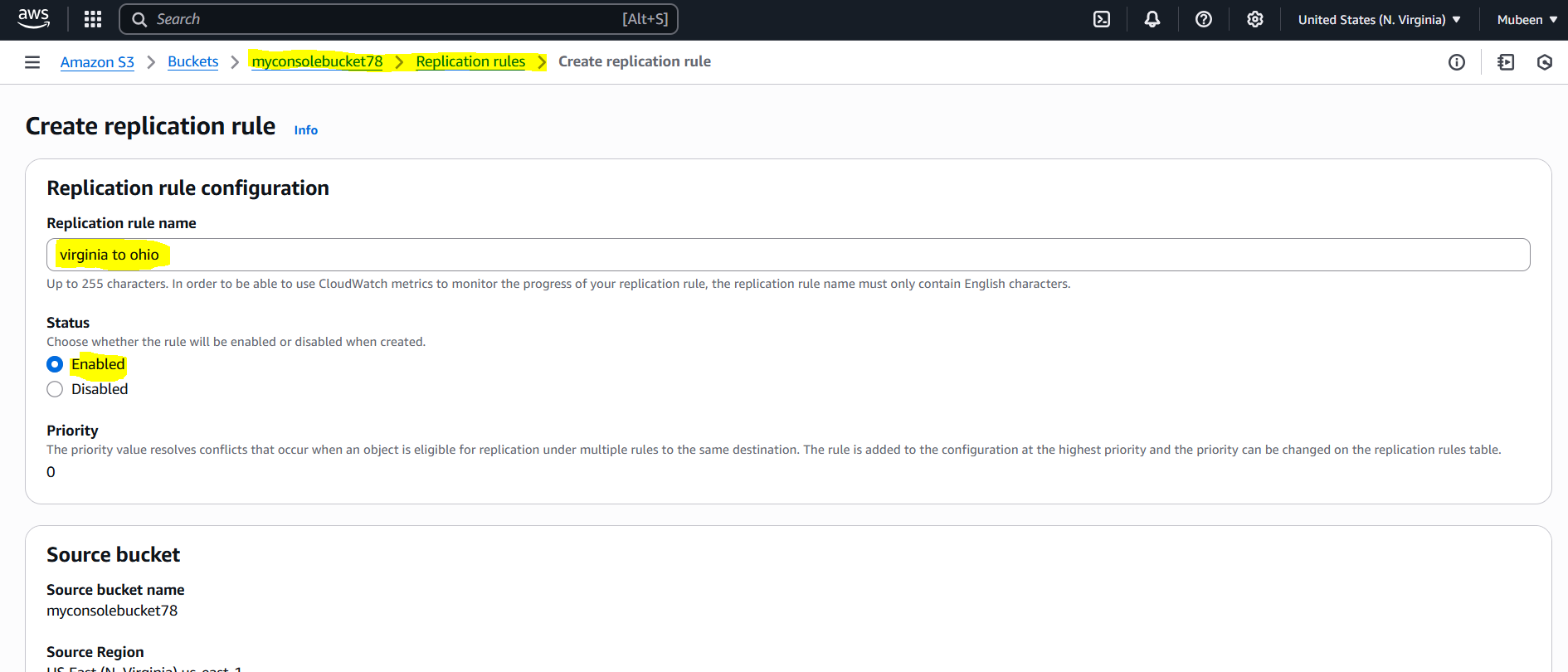
3. Click on "Replication" and then click on "Create replication rule".

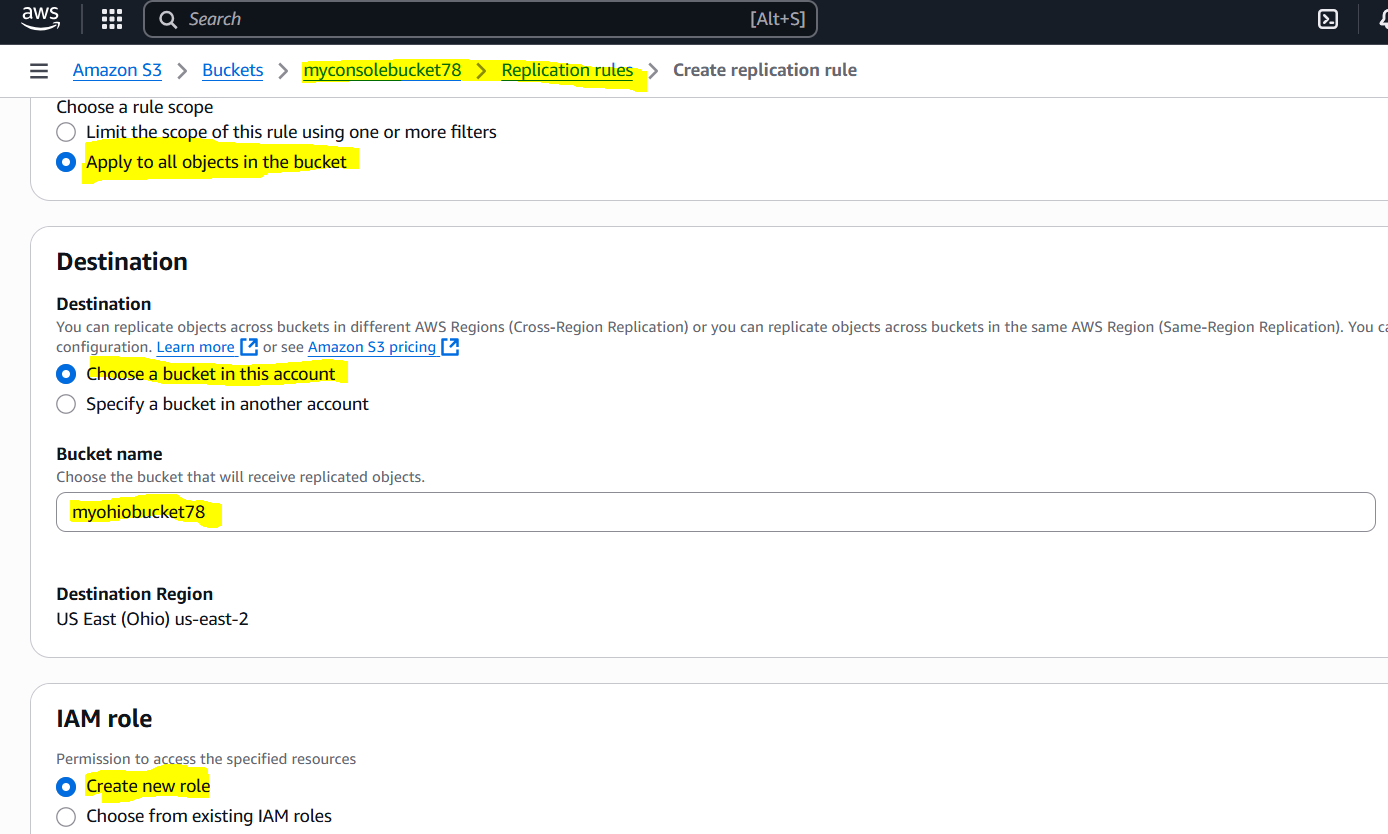
4. Select "Cross-region replication" and choose the destination bucket.

5. Choose the IAM role (create a new role for you).

6. Configure any additional settings as needed.

7. Click "Save rule".





Policy added to IAM role of the Bucket Replication

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"s3:GetObject",

"s3:GetObjectVersion",

"s3:GetObjectVersionAcl",

"s3:ListBucket"

],

"Resource": [

"arn:aws:s3:::myconsolebucket78",

"arn:aws:s3:::myconsolebucket78/\*"

]

},

{

"Effect": "Allow",

"Action": [

"s3:ReplicateObject",

"s3:ReplicateDelete"

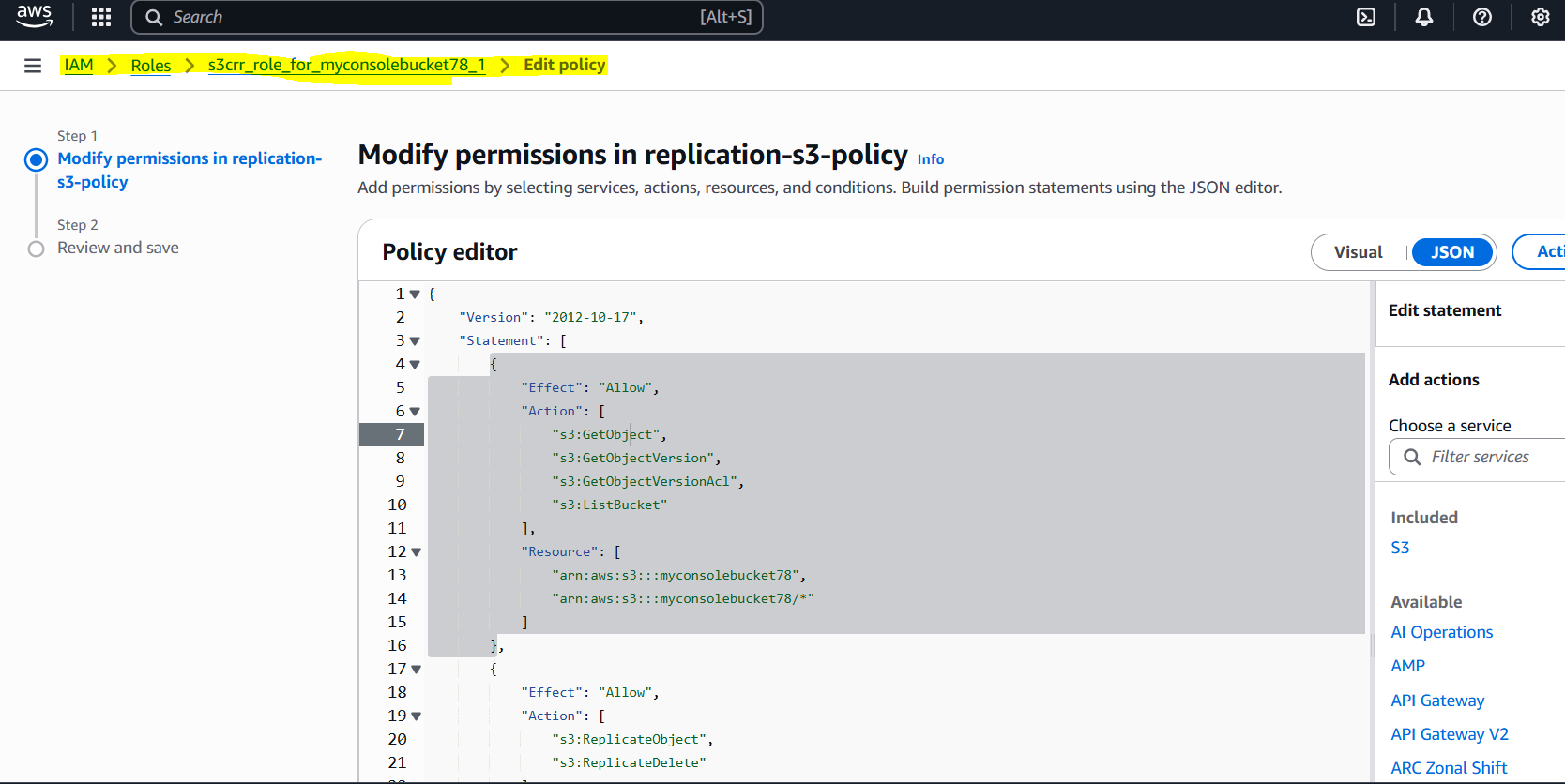
],

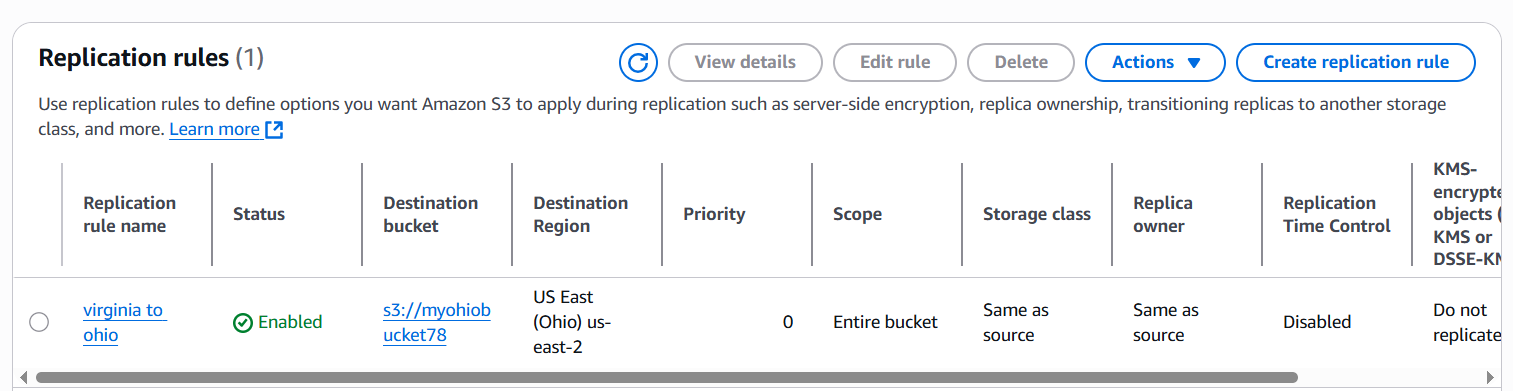
"Resource": "arn:aws:s3:::myohiobucket78/\*"

}

]

}



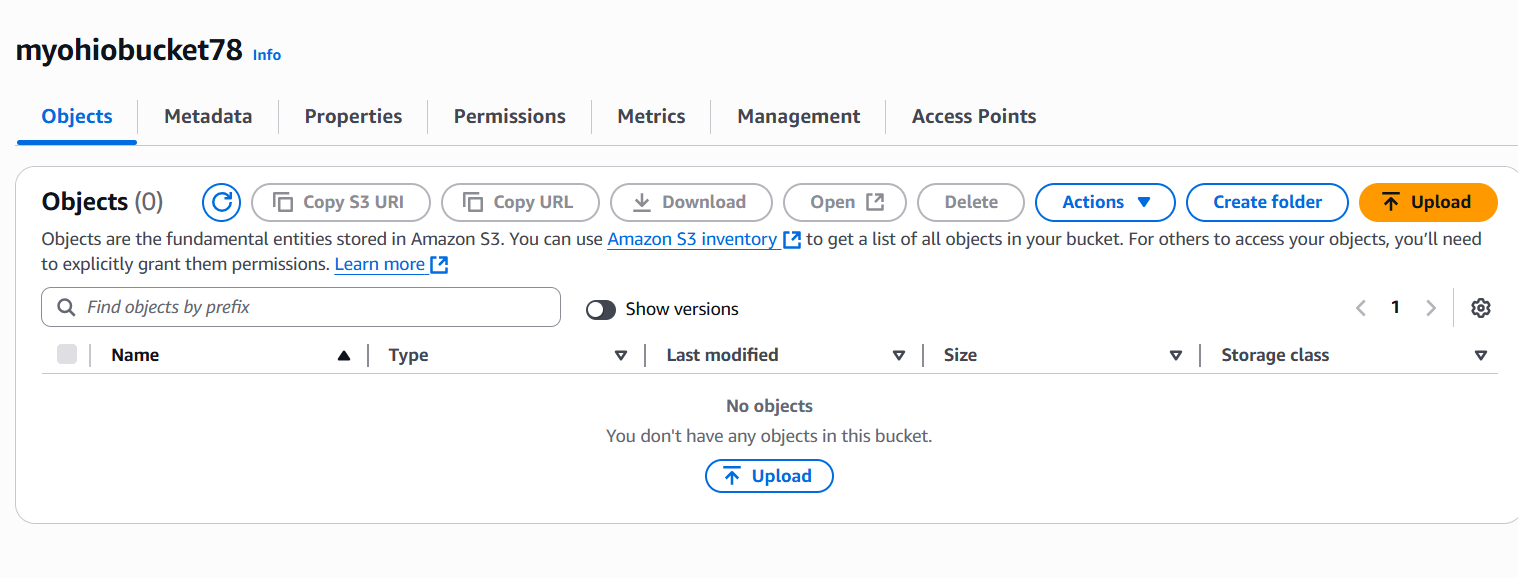


**Step 3: Verify Replication**

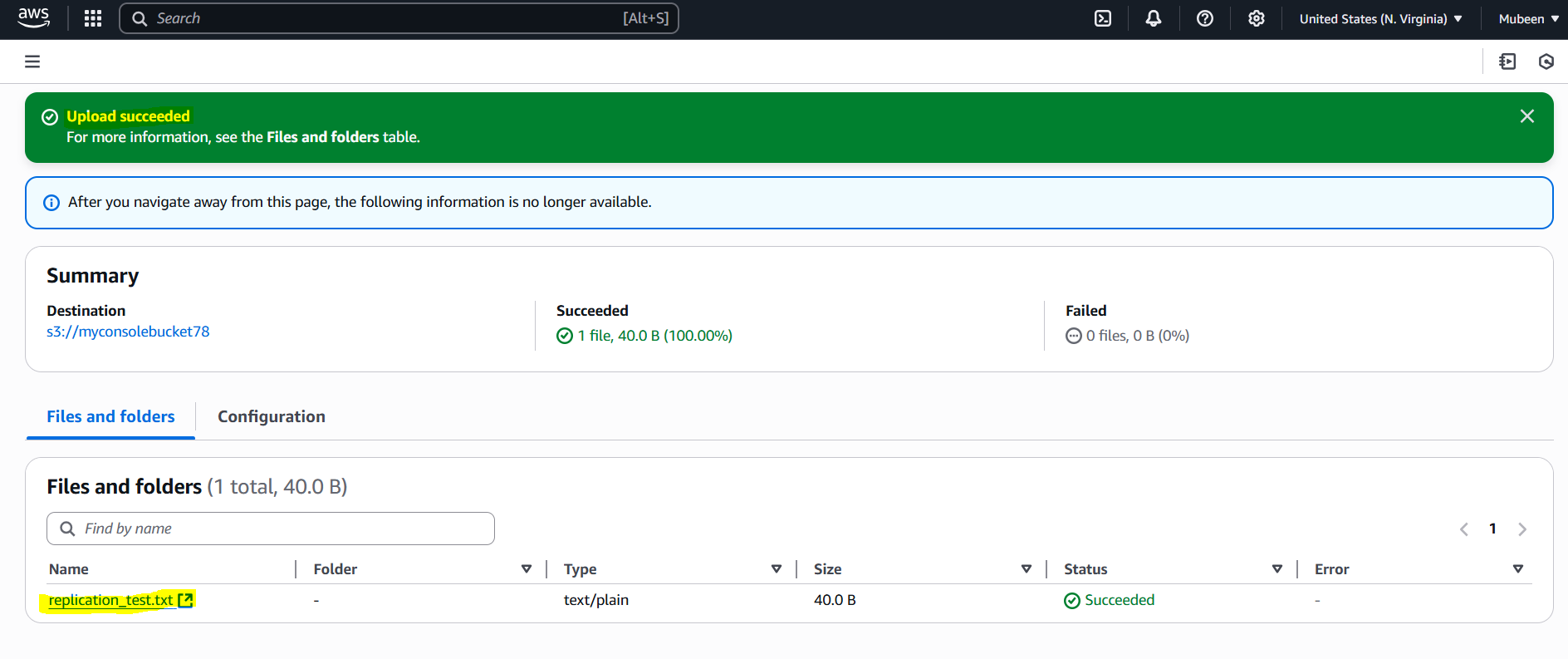
1. Upload a test object to the source bucket.

2. Go to the destination bucket and verify that the object has been replicated.

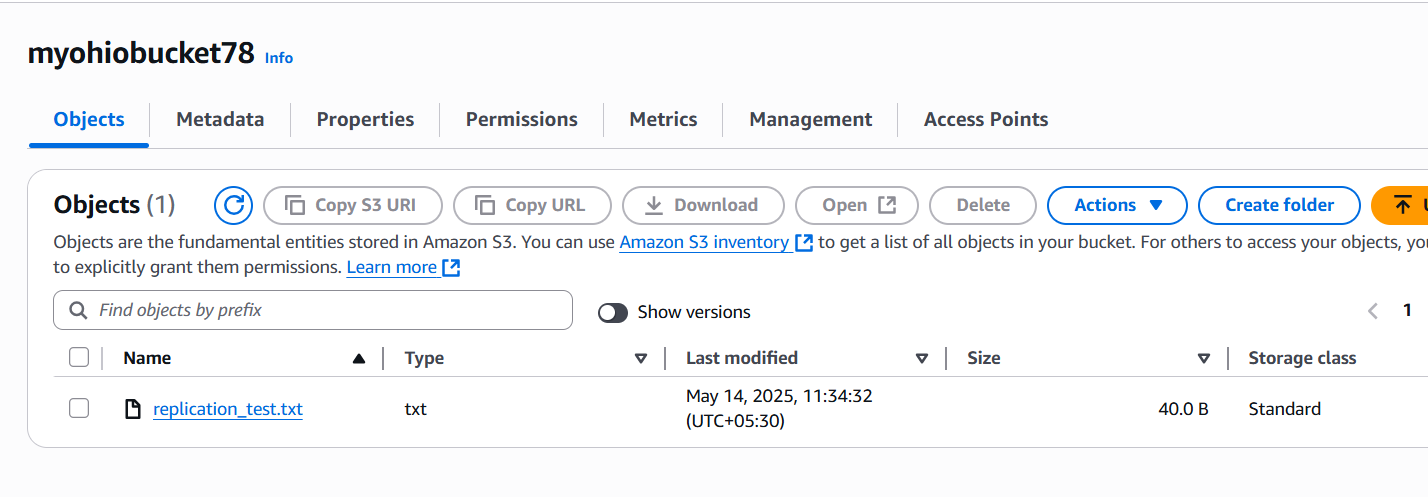
We can see that we don’t have any objects in ohio region bucket.



File added in to the bucket **myconsolebucket78.**



File is replicated in to the bucket **myohiobucket78 (ohio).**

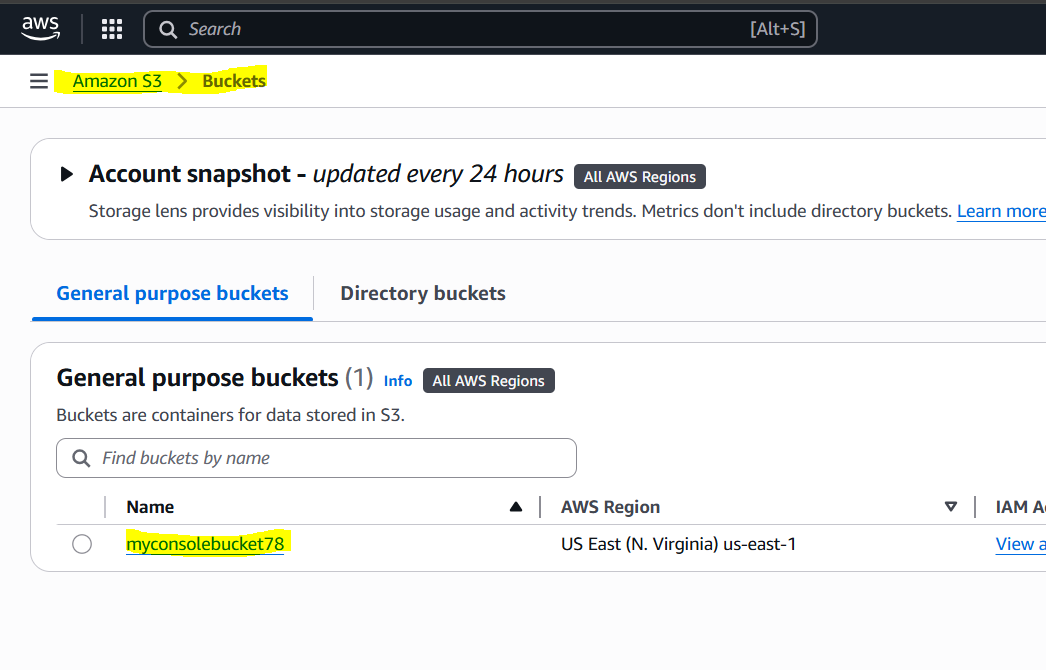


**4) Configure bucket policy, only Admin user can see the objects of s3 bucket.   
  
Step 1: Select the Bucket**

1. Log in to the AWS Management Console.

2. Navigate to the S3 dashboard.

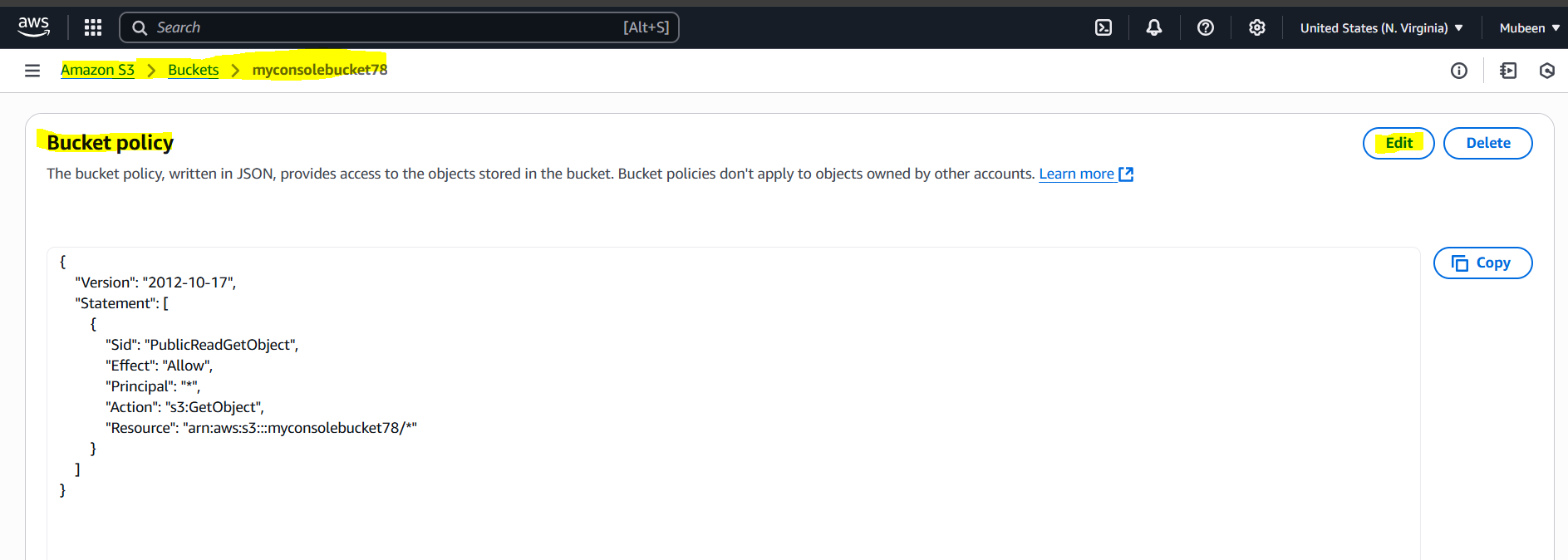
3. Select the bucket (e.g., myconsolebucket78).



**Step 2: Edit Bucket Policy**

1. Click "Permissions" and then "Bucket policy".

2. Click "Edit".



**Step 3: Configure and save the Policy**

1. In the policy document, add the following

2. Click "Save changes" to apply the bucket policy.

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "AllowOnlyRootAccess",

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::800480363349:root"

},

"Action": "s3:\*",

"Resource": [

"arn:aws:s3:::myconsolebucket78",

"arn:aws:s3:::myconsolebucket78/\*"

]

},

{

"Sid": "DenyAllOthers",

"Effect": "Deny",

"Principal": "\*",

"Action": "s3:\*",

"Resource": [

"arn:aws:s3:::myconsolebucket78",

"arn:aws:s3:::myconsolebucket78/\*"

],

"Condition": {

"StringNotEquals": {

"aws:PrincipalArn": "arn:aws:iam::800480363349:root"

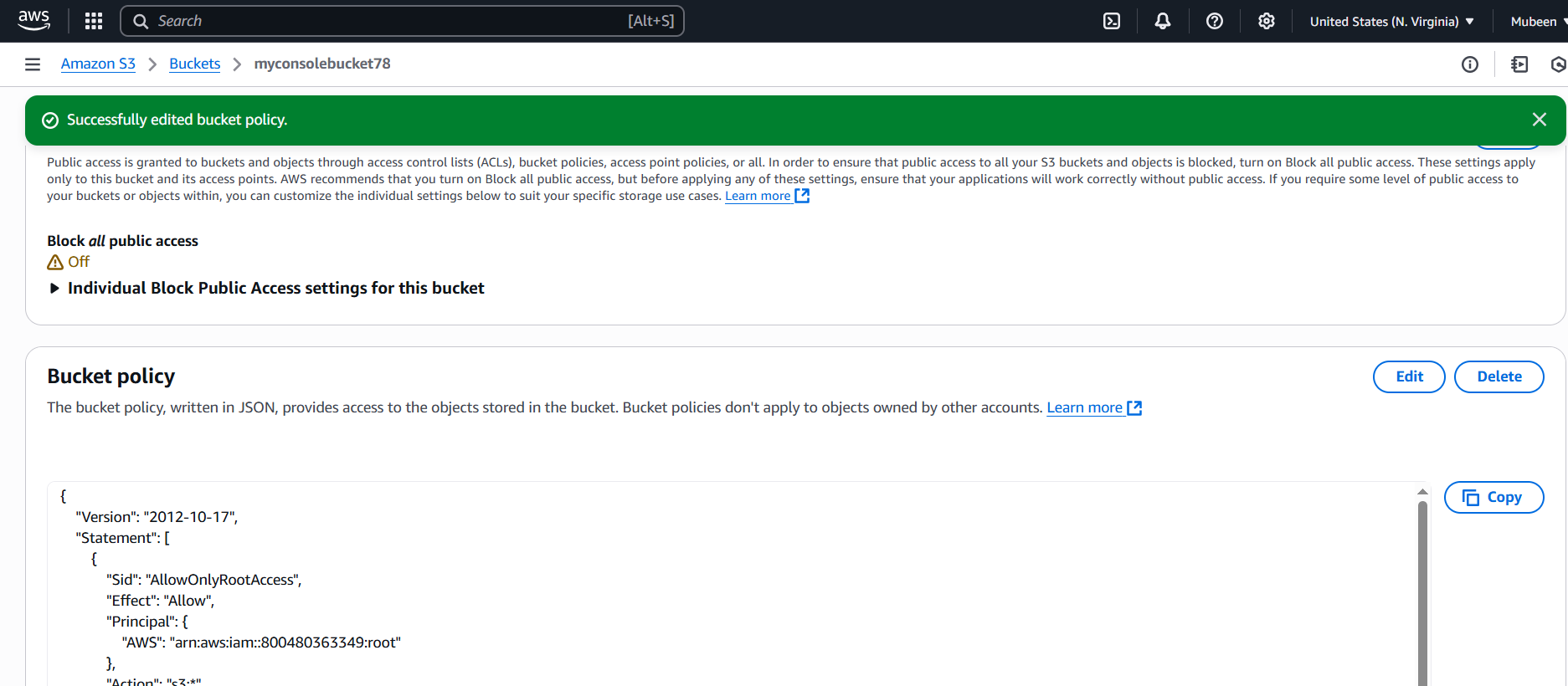
}

}

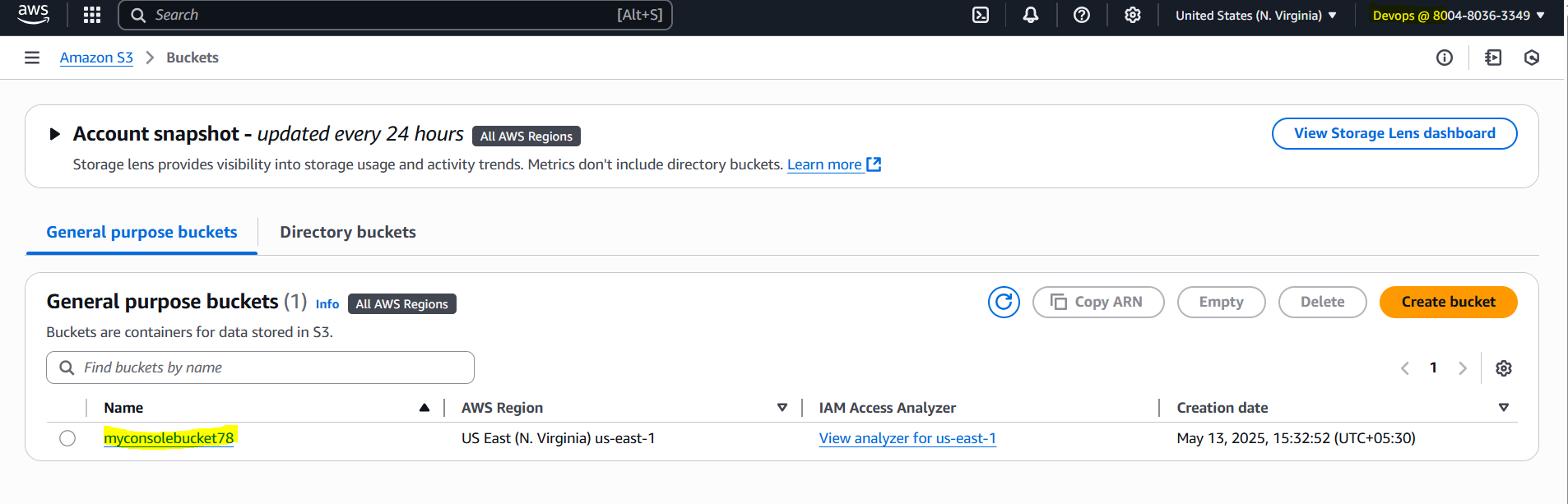
}

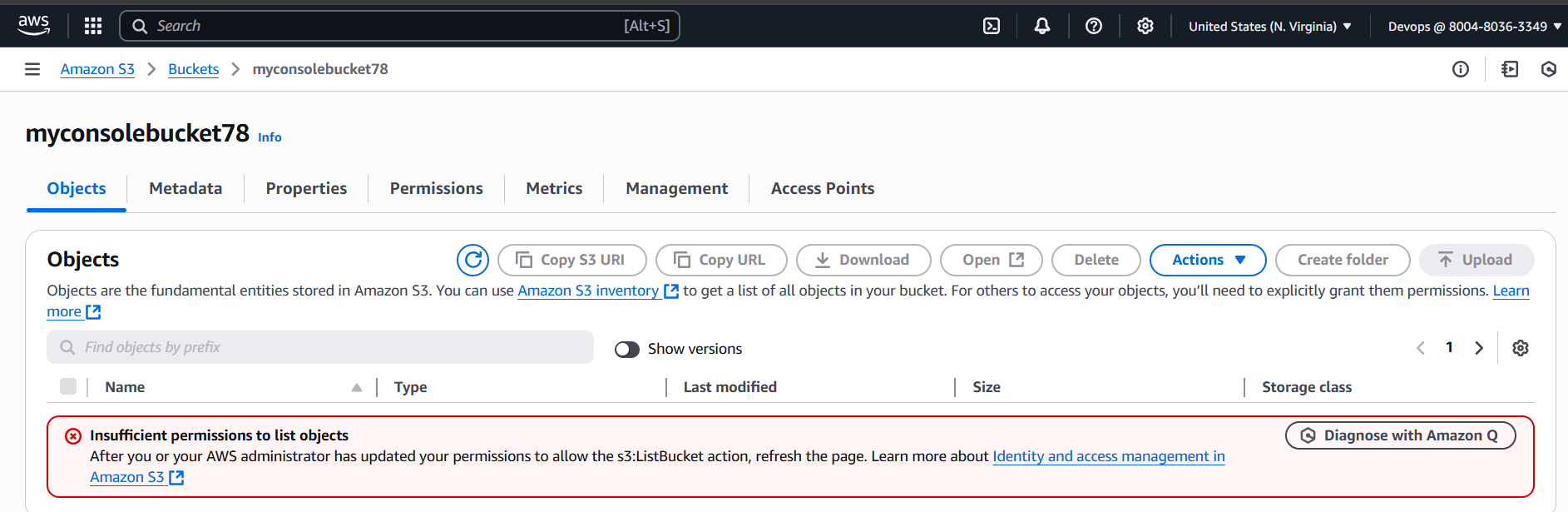
]

}



We can see that other IAM user devops can see the bucket but don’t have access for objects.  
Only Admin/root user have all access for objects now.



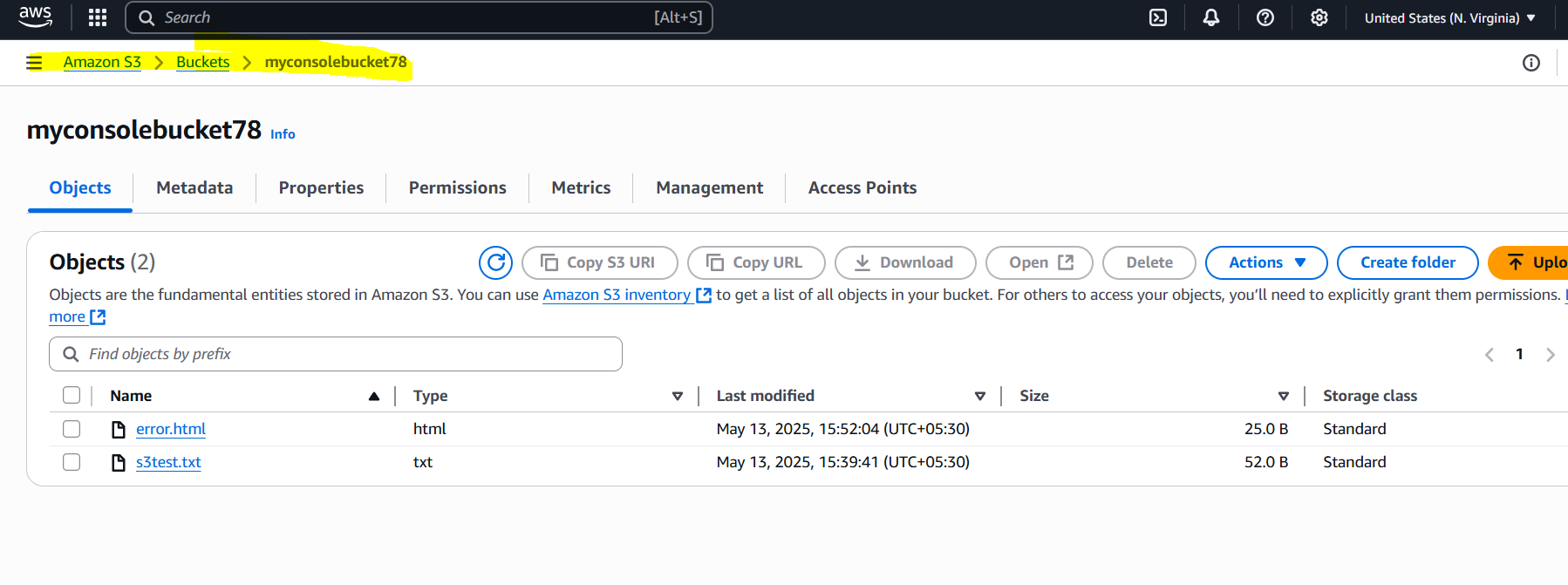


**5) Setup lifecycle policies to automatically transition or delete objects based on specific criteria.   
  
Step 1: Navigate to the S3 Dashboard**

1. Log in to the AWS Management Console.

2. Navigate to the S3 dashboard.

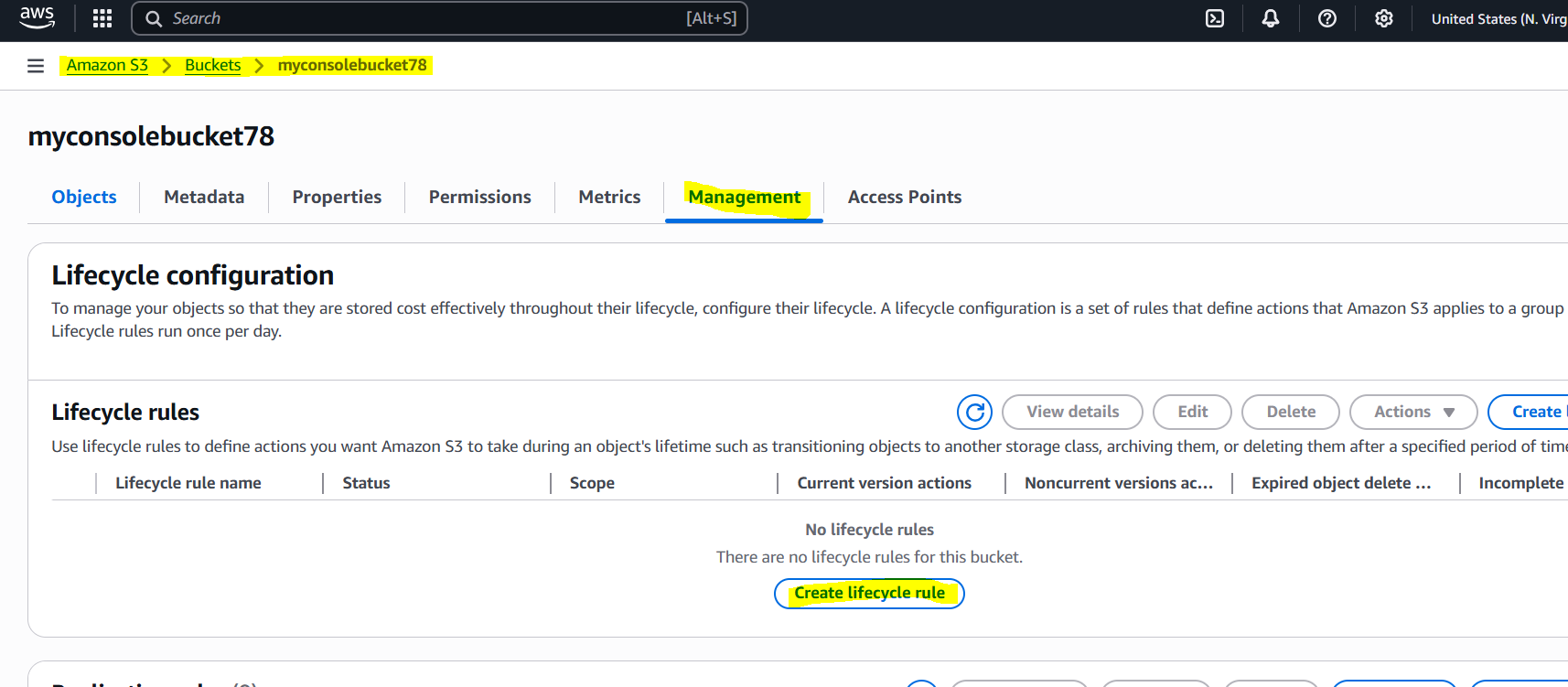
3. Select the bucket (e.g., myconsolebucket78) for which you want to set up lifecycle policies.



**Step 2: Open Lifecycle Configuration**

1. Click "management" and then "Lifecycle configuration".

2. Click "Create lifecycle rule".



**Step 3: Configure Lifecycle Rule**

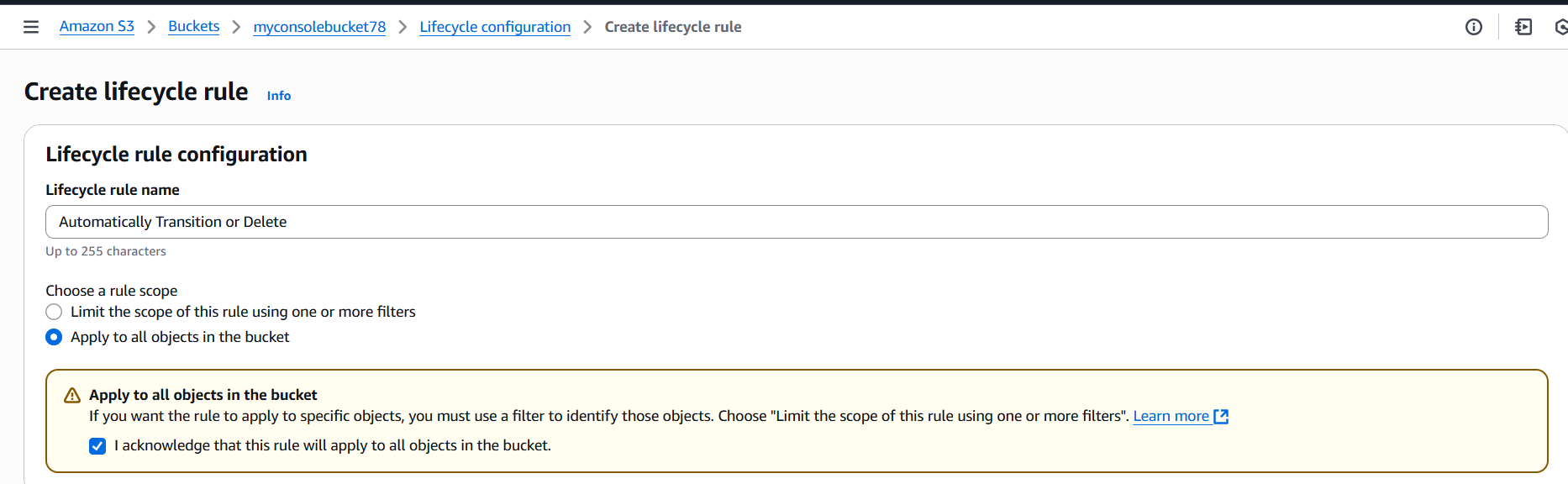
1. Enter a rule name (e.g., "TransitionToGlacier").

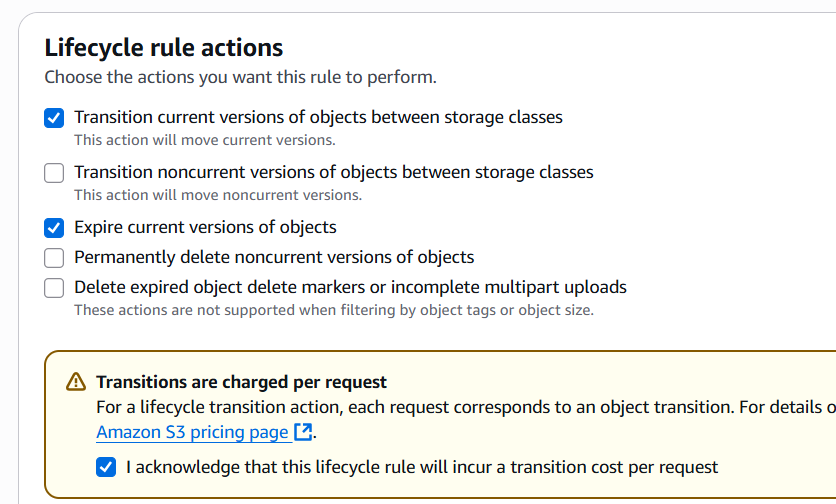
2. Choose the scope of the rule (e.g., "Whole bucket" or "Prefix").

3. 3. Under Lifecycle rule actions, select:

- Transition current versions of objects between storage classes

- Expire current versions of objects





**Step 4: Set Transition or Expiration Actions**

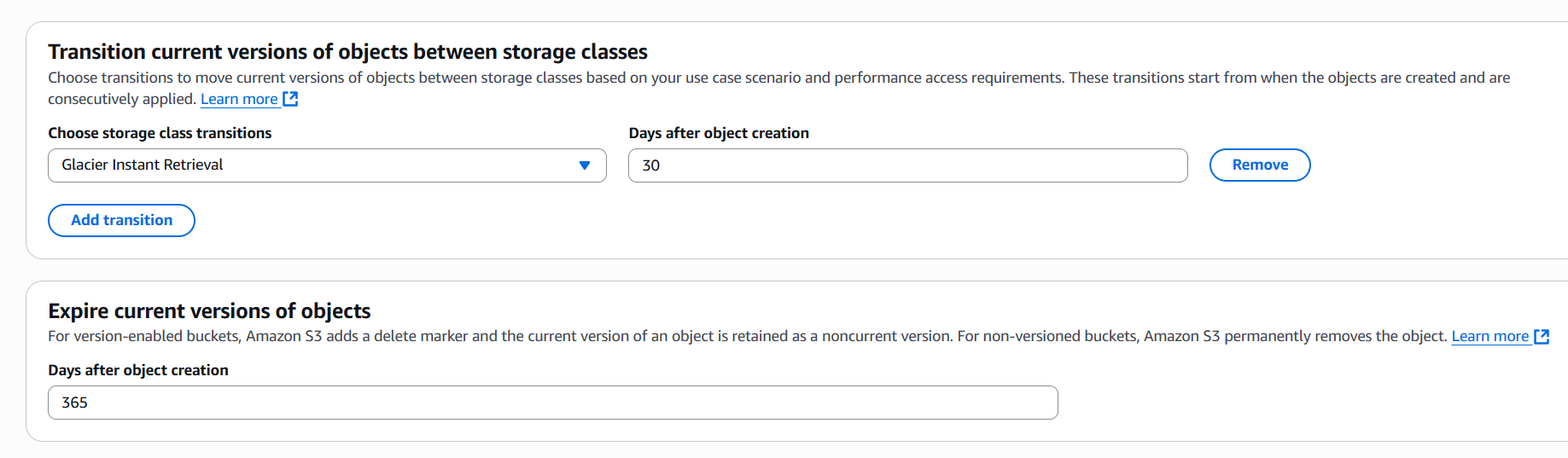
1. For "Transition current versions of objects between storage classes":

- Select the storage class (e.g., Glacier).

- Set the number of days after which objects should transition (e.g., 30).

2. For "Expire current versions of objects":

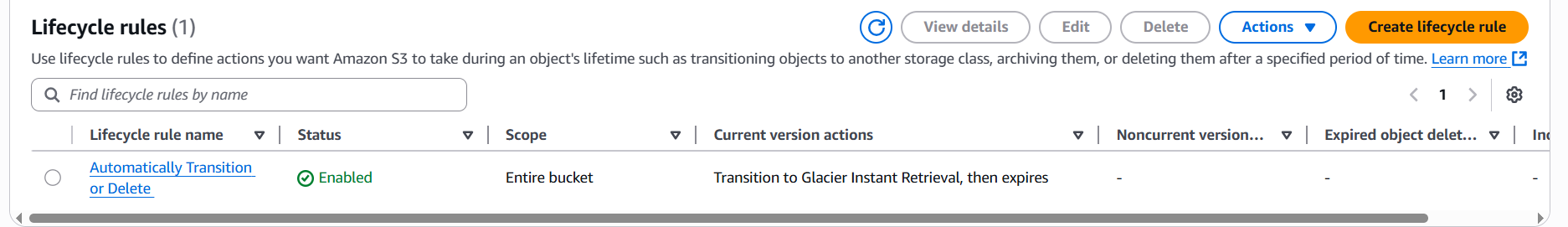
- Set the number of days after which objects should expire (e.g., 365)



**Step 5: Review and Save the Rule**

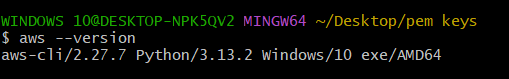
1. Review the lifecycle rule settings.

2. Click "Save rule" to apply the lifecycle configuration.

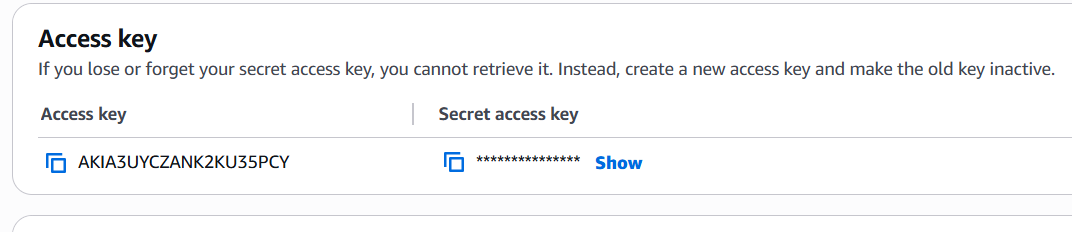


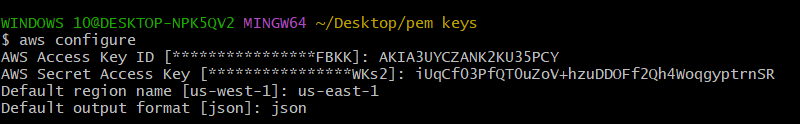
**6) Push some objects in s3 using AWS CLI.   
  
Step 1: Install and Configure AWS CLI**

1. Ensure you have AWS CLI installed on your machine, using command **aws --version**.

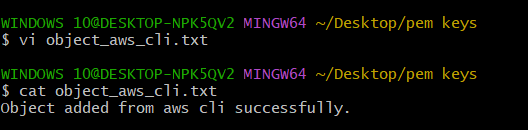


2. Run aws configure to set up your AWS credentials, using command aws configure.



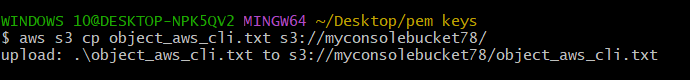


**Step 2: Create a File to Upload**

1. Create a file that you want to upload to S3.  
  


**Step 3: Upload the File to S3**

1. Use the following command to upload the file:

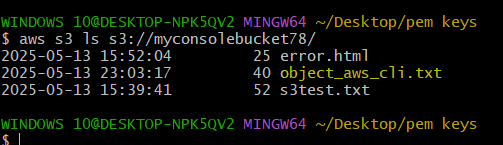


**aws s3 cp object\_aws\_cli.txt s3://myconsolebucket78/**

We are using cp command with file name and bucket name.

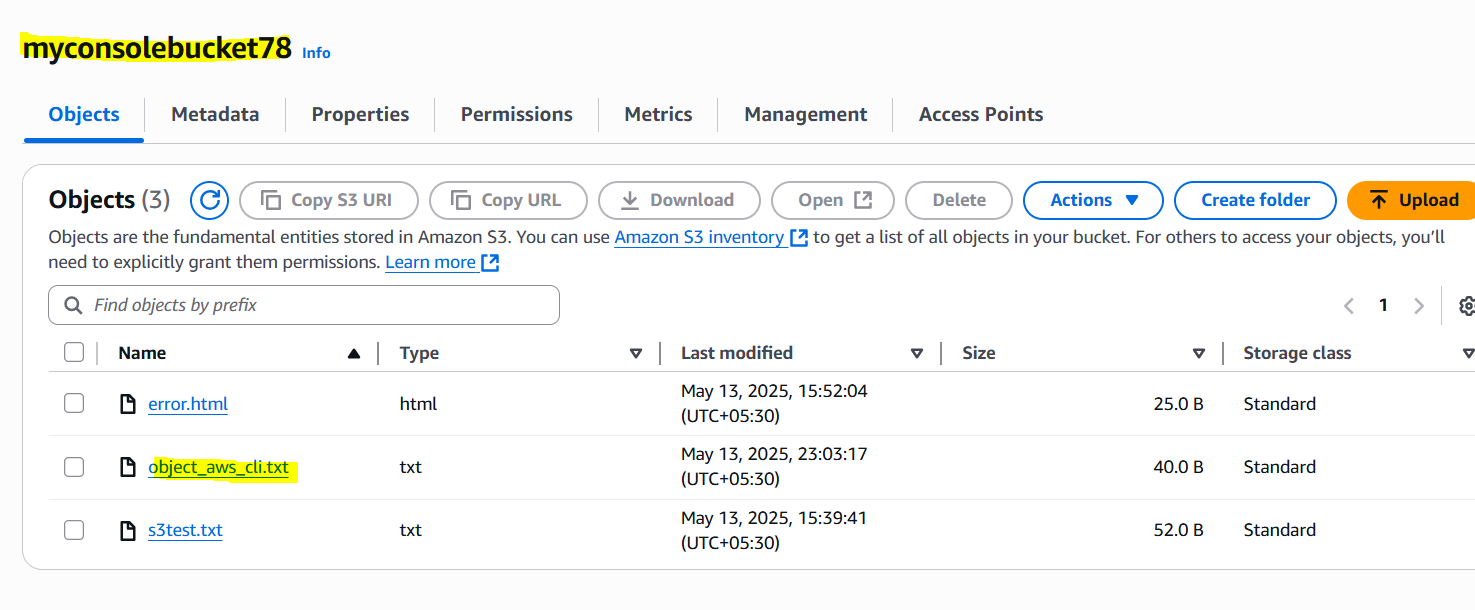
**Step 4: Verify the Upload**

1. Use the following command to list the objects in your bucket:



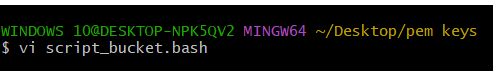
**aws s3 ls s3:// myconsolebucket78/**

This should display the file you just uploaded.

We can also check at console.  
  


**7) Write a bash script to create s3 bucket.   
  
Step 1: Create a Bash Script**

1. Open a text editor and create a new file named **script\_bucket.bash**.



**Step 2: Add the Script Content**

1. Add the following script content to the file:

**#!/bin/bash**

**# Purpose: To create a S3 Bucket**

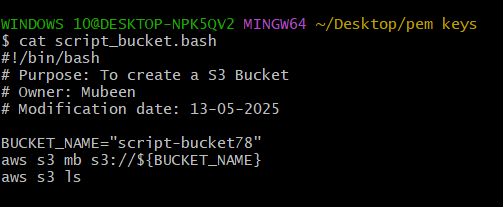
**# Owner: Mubeen**

**# Modification date: 13-05-2025**

**BUCKET\_NAME="script-bucket"**

**aws s3 mb s3://${BUCKET\_NAME}**

**aws s3 ls**



**Step 3: Make the Script Executable**

1. Run the following command to make the script executable:

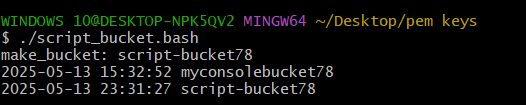
**chmod +x script\_bucket.bash**

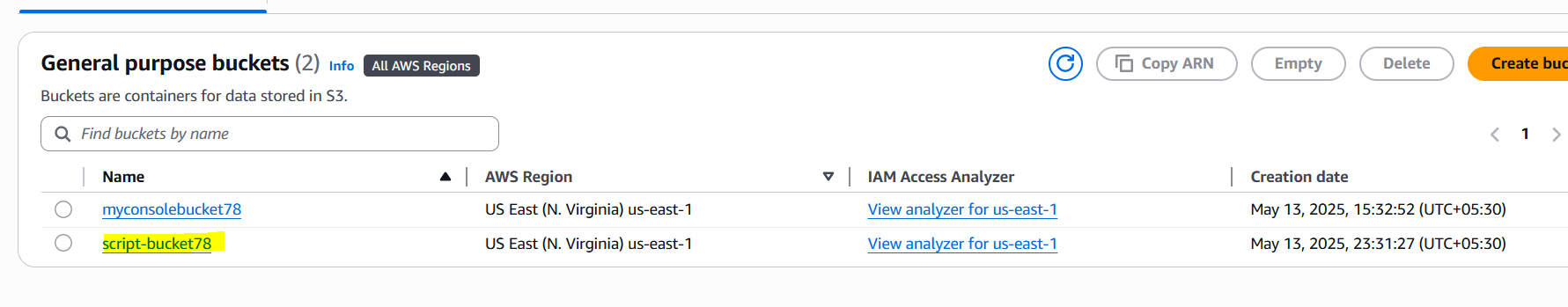


**Step 4: Run the Script**

1. Run the script using the following command:

**./script\_bucket.bash**



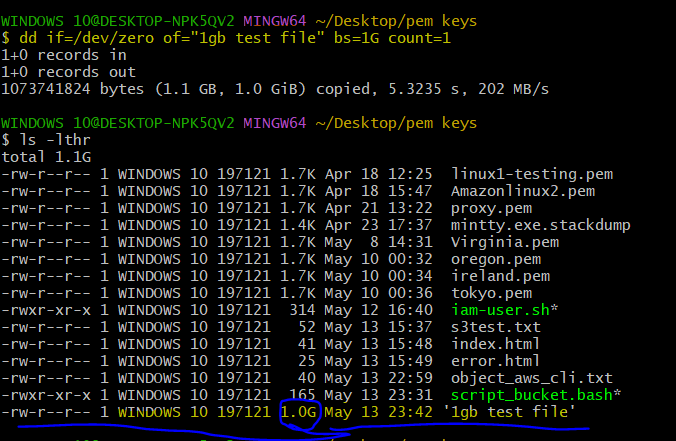
We can also confirm with console.  
  


**8) Upload one 1 gb of file to s3 using cli.**

**Step 1: Create a 1 GB Test File**

1. Open your terminal and run the following command:

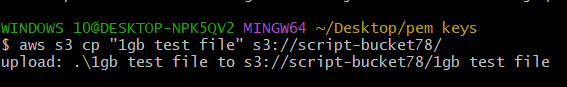
**dd if=/dev/zero of="1gb test file" bs=1G count=1**



This will create a 1 GB file named "1gb test file" in your current directory.

**Step 2: Upload the File to S3**

1. Run the following command to upload the file to your S3 bucket:



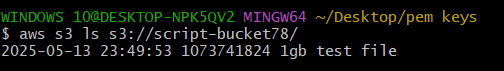
**aws s3 cp "1gb test file" s3://script-bucket78/**

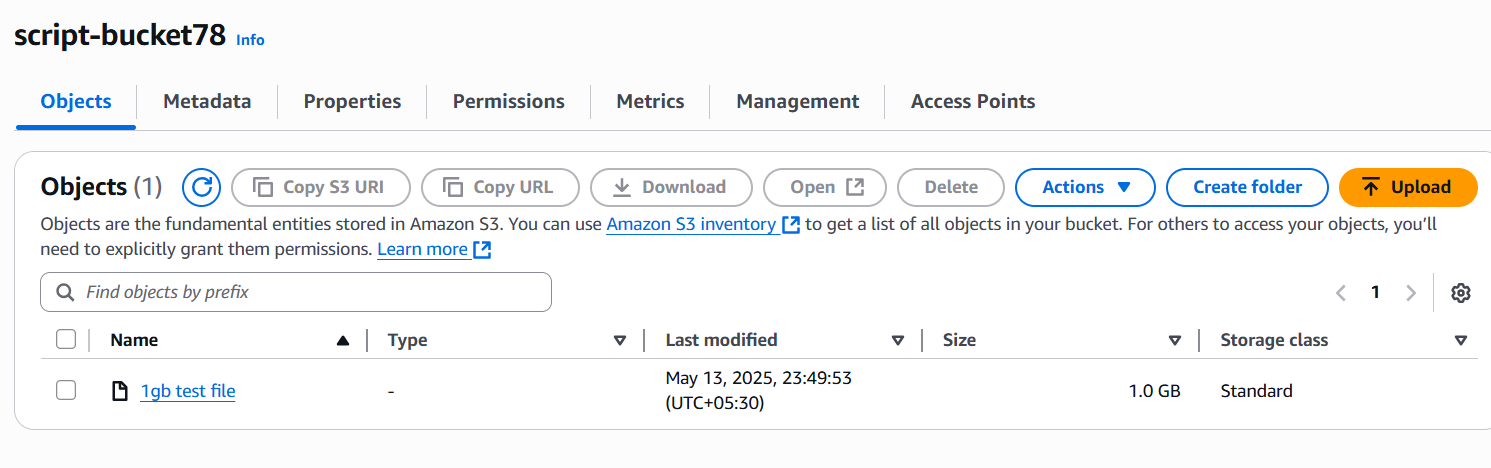
This will copy the file to your S3 bucket named "script-bucket78".

**Step 3: Verify the Upload**

1. Run the following command to verify that the file was uploaded successfully:

**aws s3 ls s3://script-bucket78/**



We can also verify at console.  


**The-End**