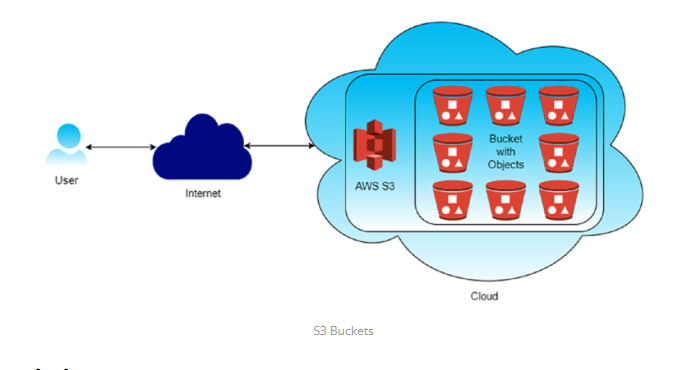
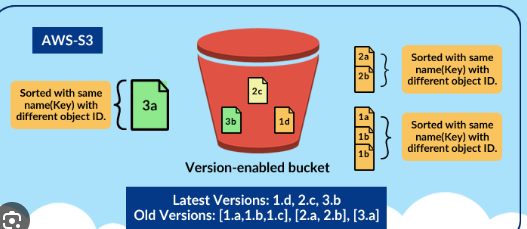
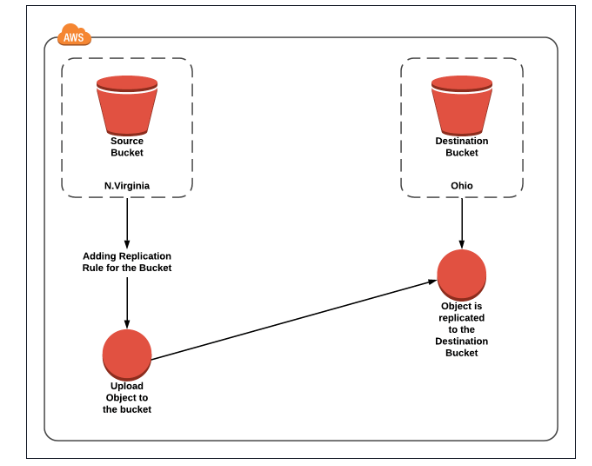
**Simple Storage Service (S3)**

**1. Versioning**: Versioning in Amazon S3 is a feature that allows you to keep multiple versions of an object in the same bucket. Whenever you upload a new version of an object with the same key (name) as an existing object, S3 doesn't overwrite the existing object but instead creates a new version of it. This means you can retain and access previous versions of objects, providing an additional layer of protection against accidental deletions or overwrites

1. **Accidental Deletion Protection** :: With versioning enabled, if you accidentally delete an object, you can still retrieve previous versions of it, preventing data loss.
2. **Accidental Overwrite Protection**:**:**Versioning prevents accidental overwrites of objects. Even if you upload a new version of an object with the same name, the previous versions are preserved.



1. **Cross-Region Replication**: Cross-region replication (CRR) in Amazon S3 is a feature that automatically replicates objects from one bucket in one AWS region to another bucket in a different AWS region. This provides redundancy and disaster recovery capabilities, ensuring that your data remains available even if an entire AWS region becomes unavailable.
2. **Disaster Recovery**: Cross-region replication helps ensure business continuity by replicating critical data to a different geographic region, reducing the risk of data loss due to regional disasters or outages.
3. **Low-Latency Access**: Users in different geographic regions can access data from the nearest region, reducing latency and improving performance.
4. **Compliance**: Some regulatory requirements mandate data replication across multiple geographic regions for data sovereignty and compliance reasons.
5. **Event Notifications**: S3 can trigger events (e.g., object creation, deletion) that can be routed to AWS Lambda, SNS, or SQS for processing.
6. **Integration**: S3 seamlessly integrates with other AWS services such as AWS Lambda, Amazon CloudFront, AWS Glue, Amazon Athena, and Amazon EMR, enabling you to build powerful data processing and analytics workflows.

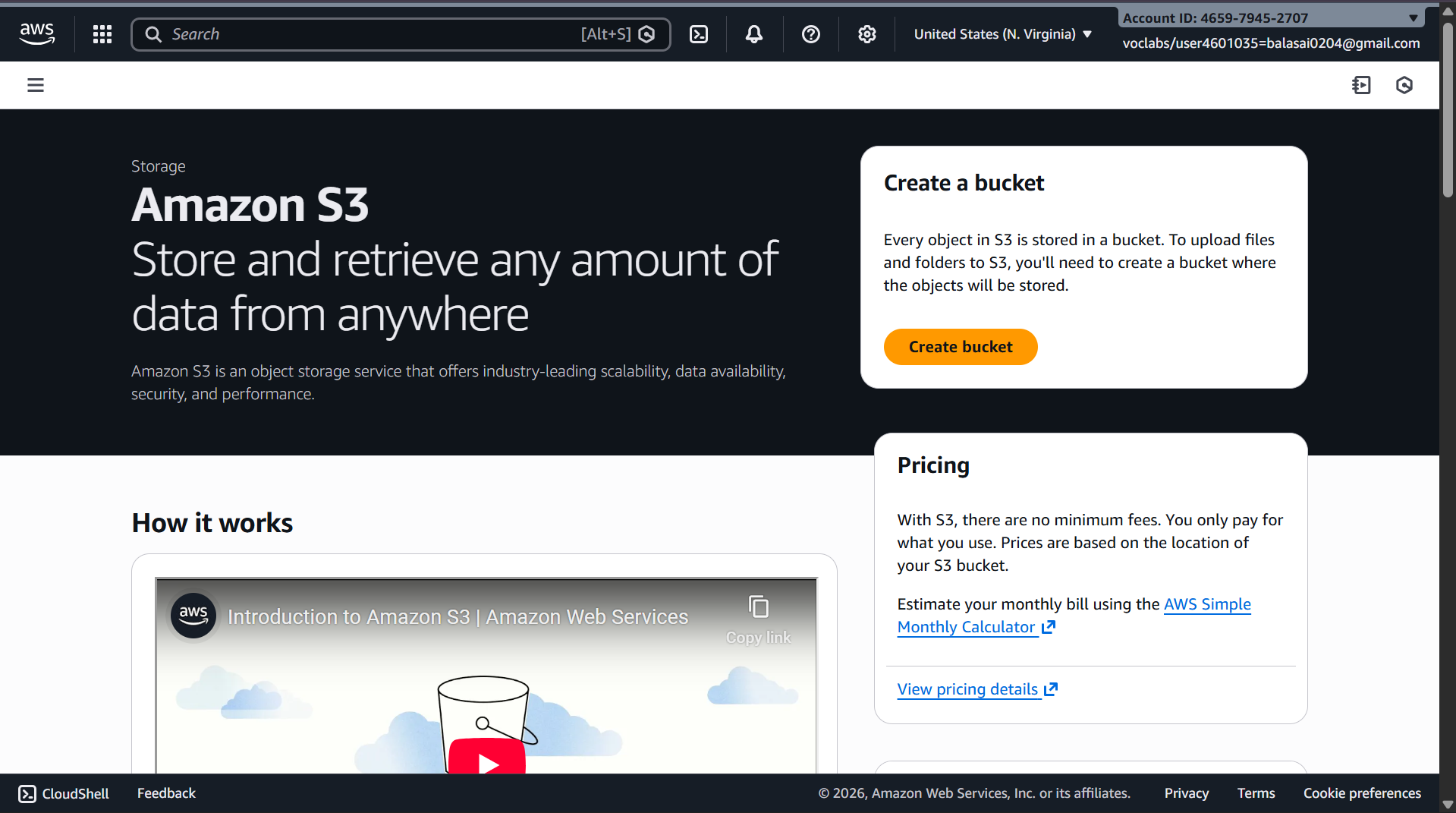


**1.Creating an S3 bucket, upload an object, and enable public access**

**1️⃣ Create an S3 Bucket (with Public Access Enabled)**

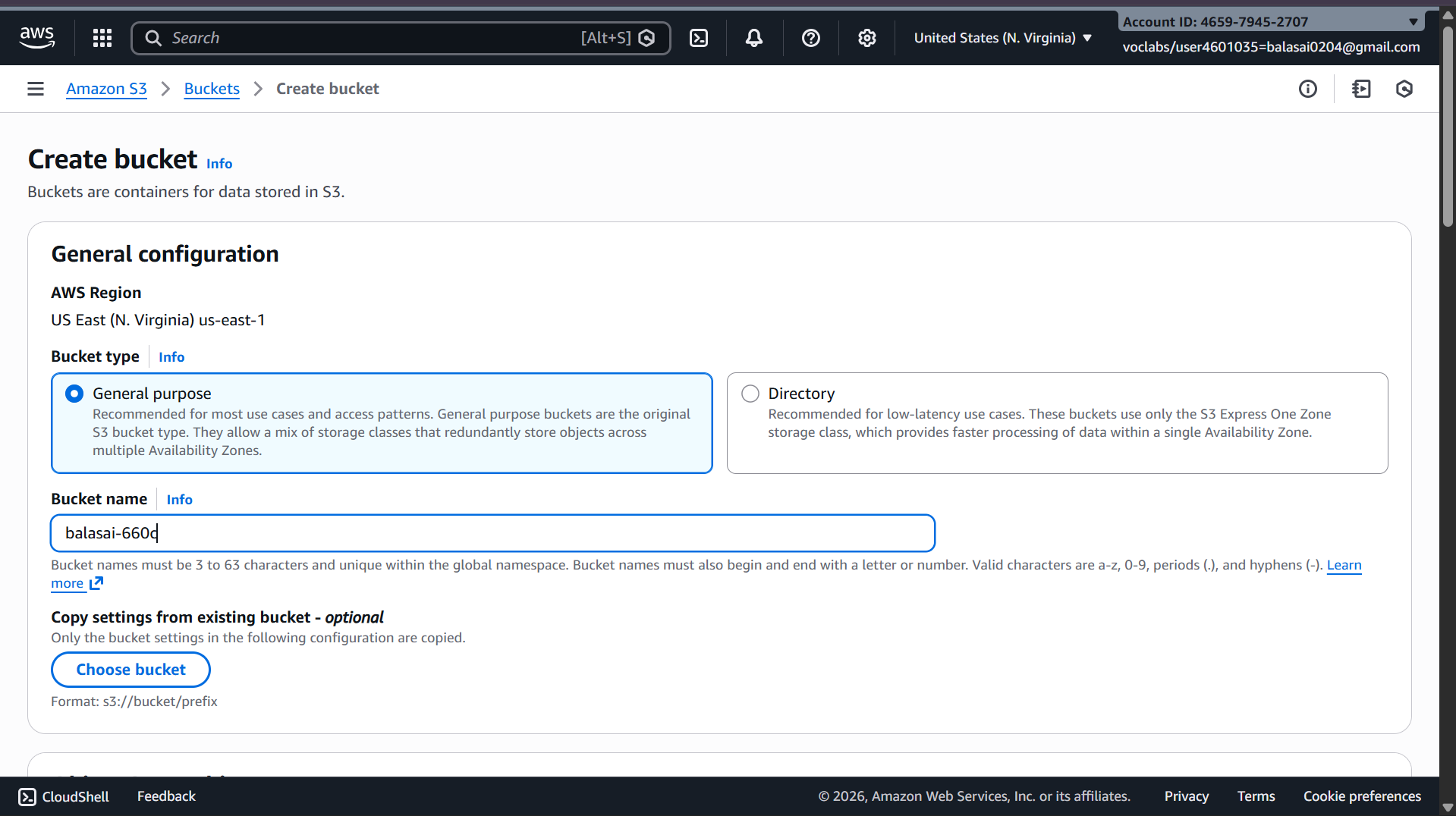
**Step 1: Open S3 Service**

* Login to AWS Management Console
* Go to Services → S3
* Click Create bucket

****

**Step 2: Bucket Configuration**

* Bucket name: my-public-bucket-123
* Region: Choose nearest region



**Step 3: Disable Block Public Access (IMPORTANT)**

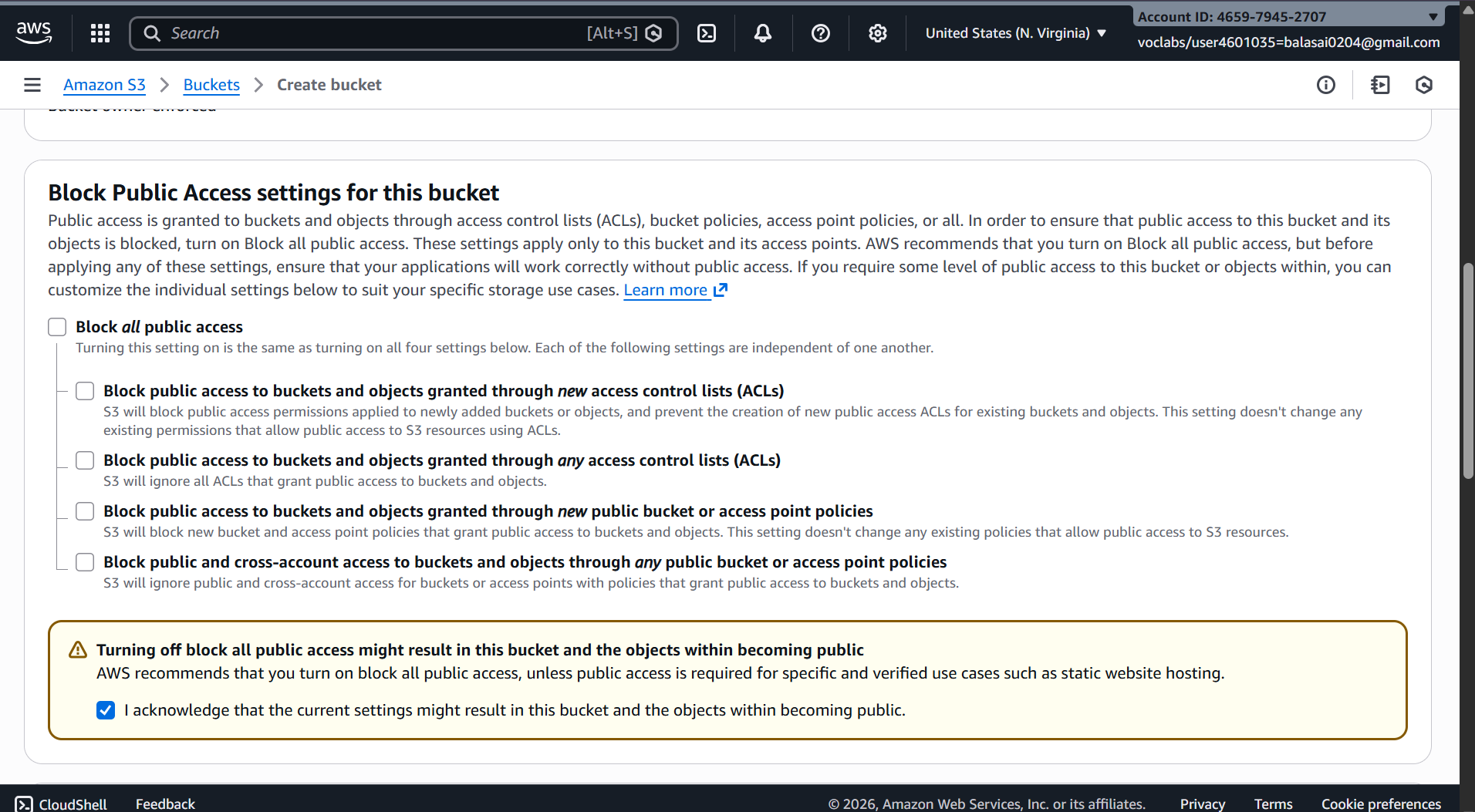
**Under Block Public Access settings for this bucket:**

**❌** Uncheck Block all public access

You must uncheck ALL options:

✔ Check I acknowledge that this bucket will become public

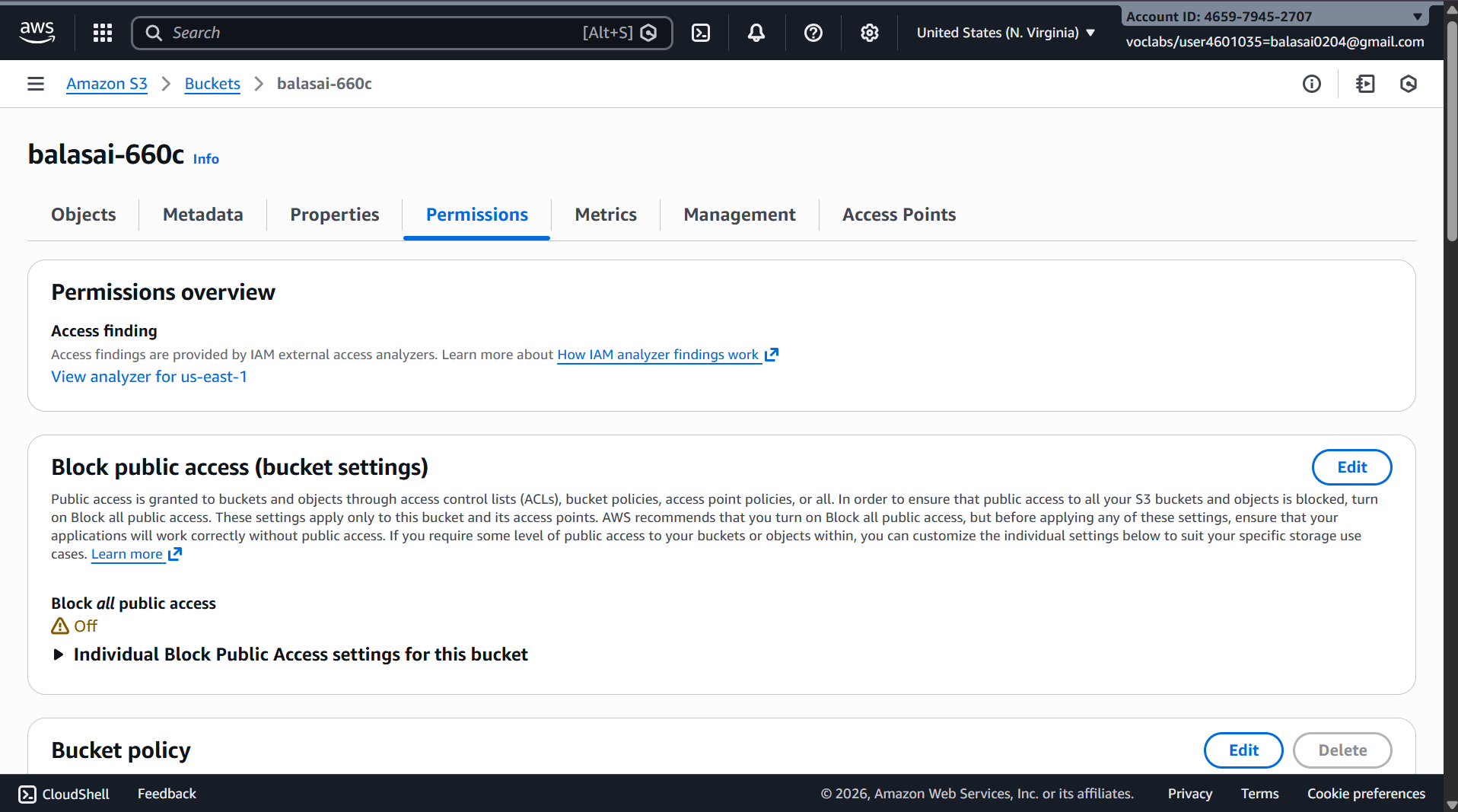
👉 Click Create bucket



**2️⃣ Enable Public Access at Bucket Level (ACL)**

**Step 4: Open the Bucket**

* Click on the bucket name
* Go to Permissions tab



**Step 5: Edit Bucket ACL**

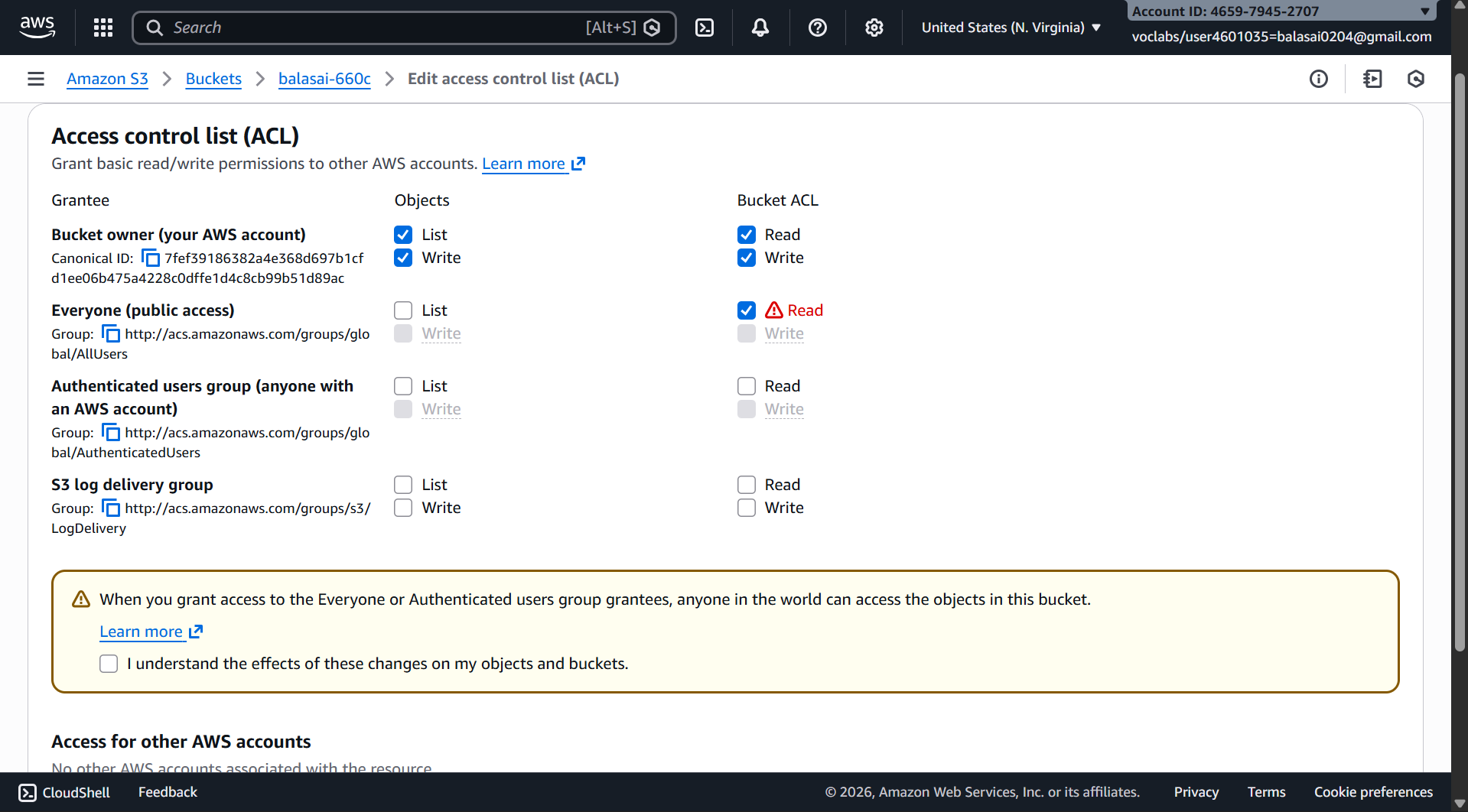
* Scroll to Access Control List (ACL)
* Click Edit

**Under Public access:**

* Everyone (public access) → ✔ Read access

**✔ Save changes**

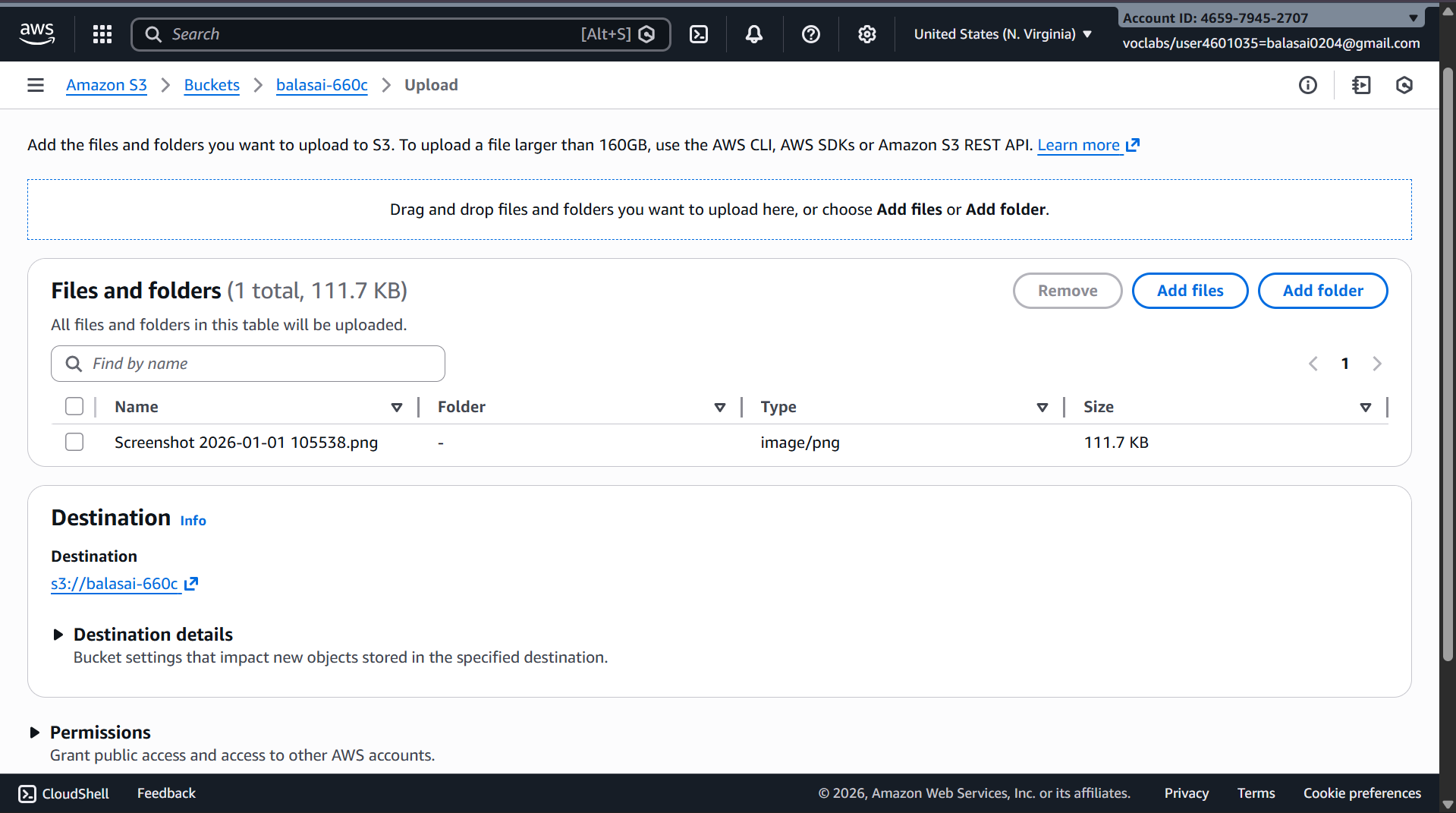
**📌 This allows public access at bucket level**

****

**3️⃣ Upload Object to the Bucket**

**Step 6: Upload File**

* Go to Objects tab
* Click Upload
* Click Add files
* Select file (e.g., image.jpg)
* Click Upload



**4️⃣ Enable Public Access for Object (ACL)**

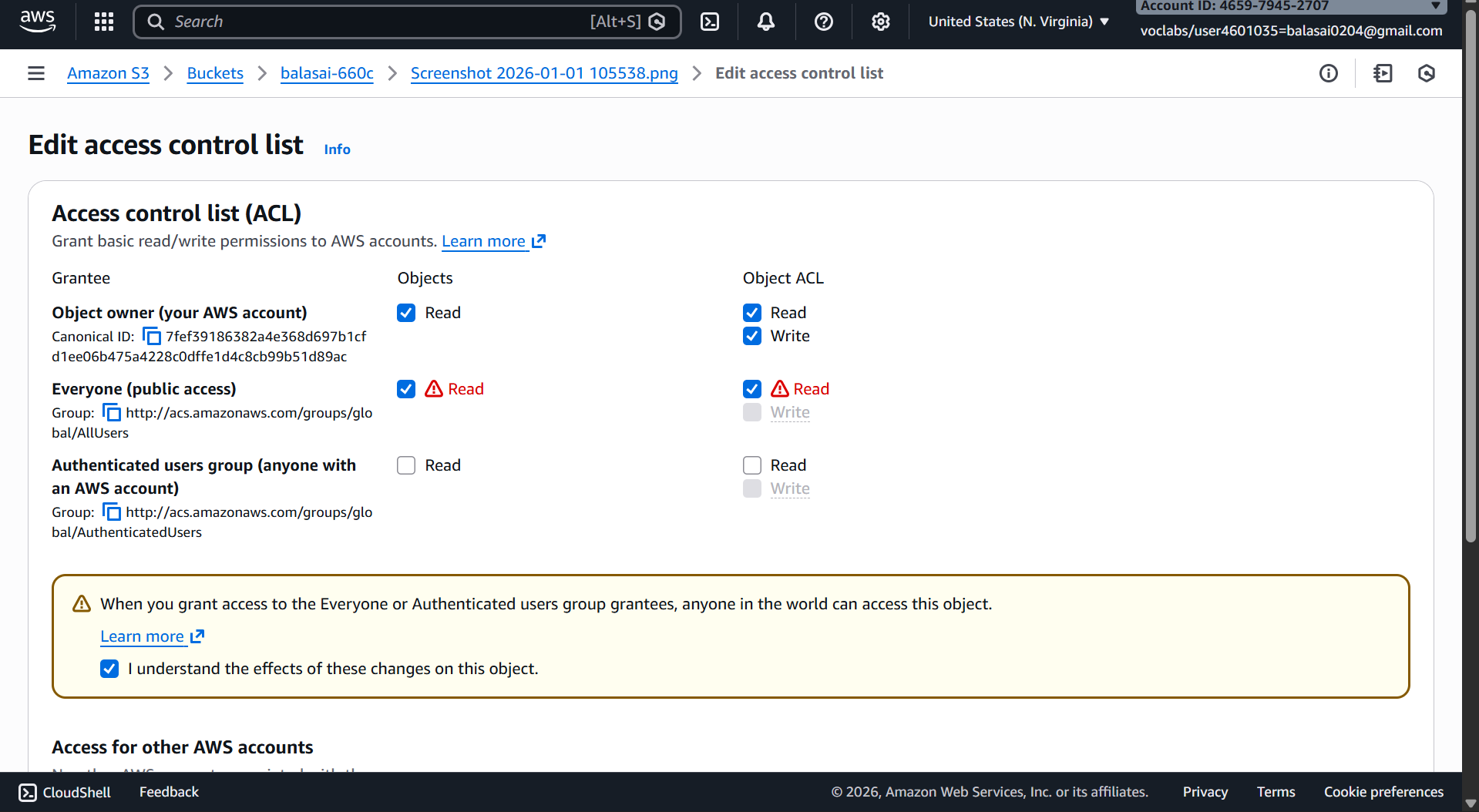
**Step 7: Object-Level ACL Permission**

* Click on uploaded object
* Go to Permissions tab
* Scroll to Access Control List (ACL)
* Click Edit

Under Public access:

* Everyone (public access) → ✔ Read access

✔ Save changes

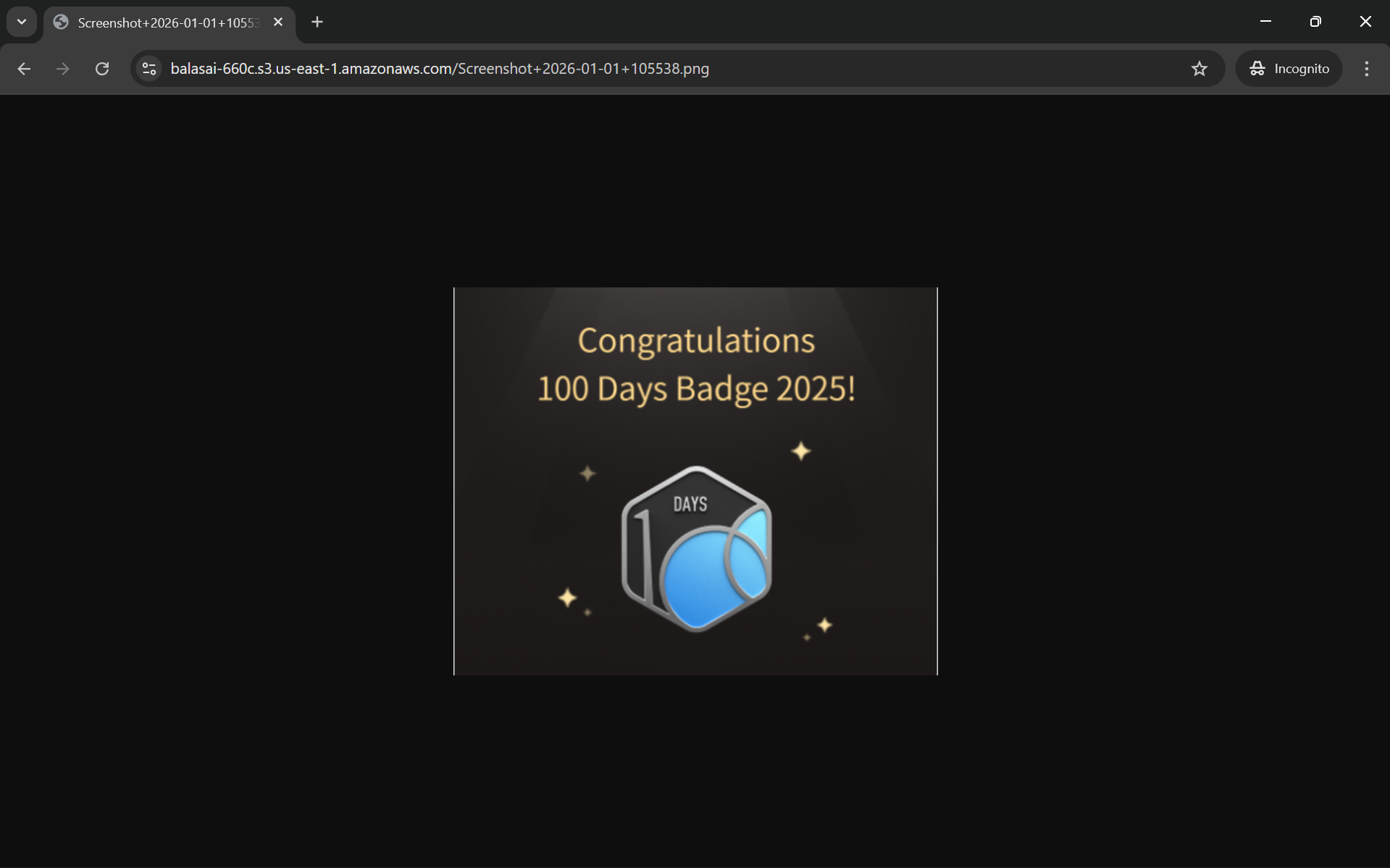


**5️⃣ Verify Public Access**

**Step 8: Test Object URL**

* Copy Object URL
* Open in browser (incognito)

✔ Object should be accessible without login



**2. S3 Versioning & Cross-Region Replication**

**1️⃣ Enabling Versioning in S3 (with Output Scenario)**

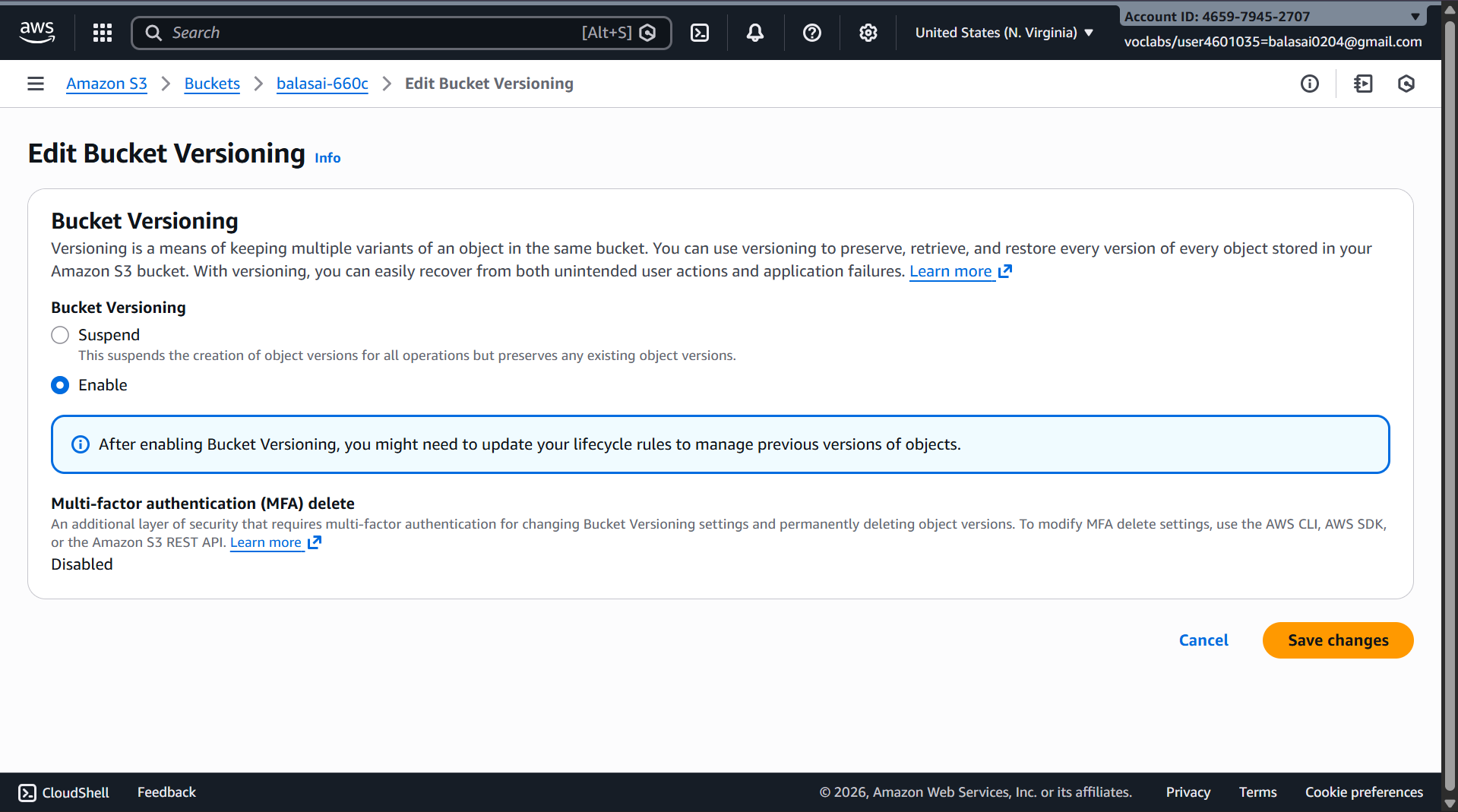
**🔸 Objective**

**To enable version control on an S3 bucket and observe object versions.**

**🔸 Steps to Enable Versioning**

1. Open AWS Management Console
2. Navigate to S3
3. Select the bucket
4. Go to Properties tab
5. Scroll to Bucket Versioning
6. Click Edit
7. Select Enable
8. Click Save changes

✔ Versioning is now enabled



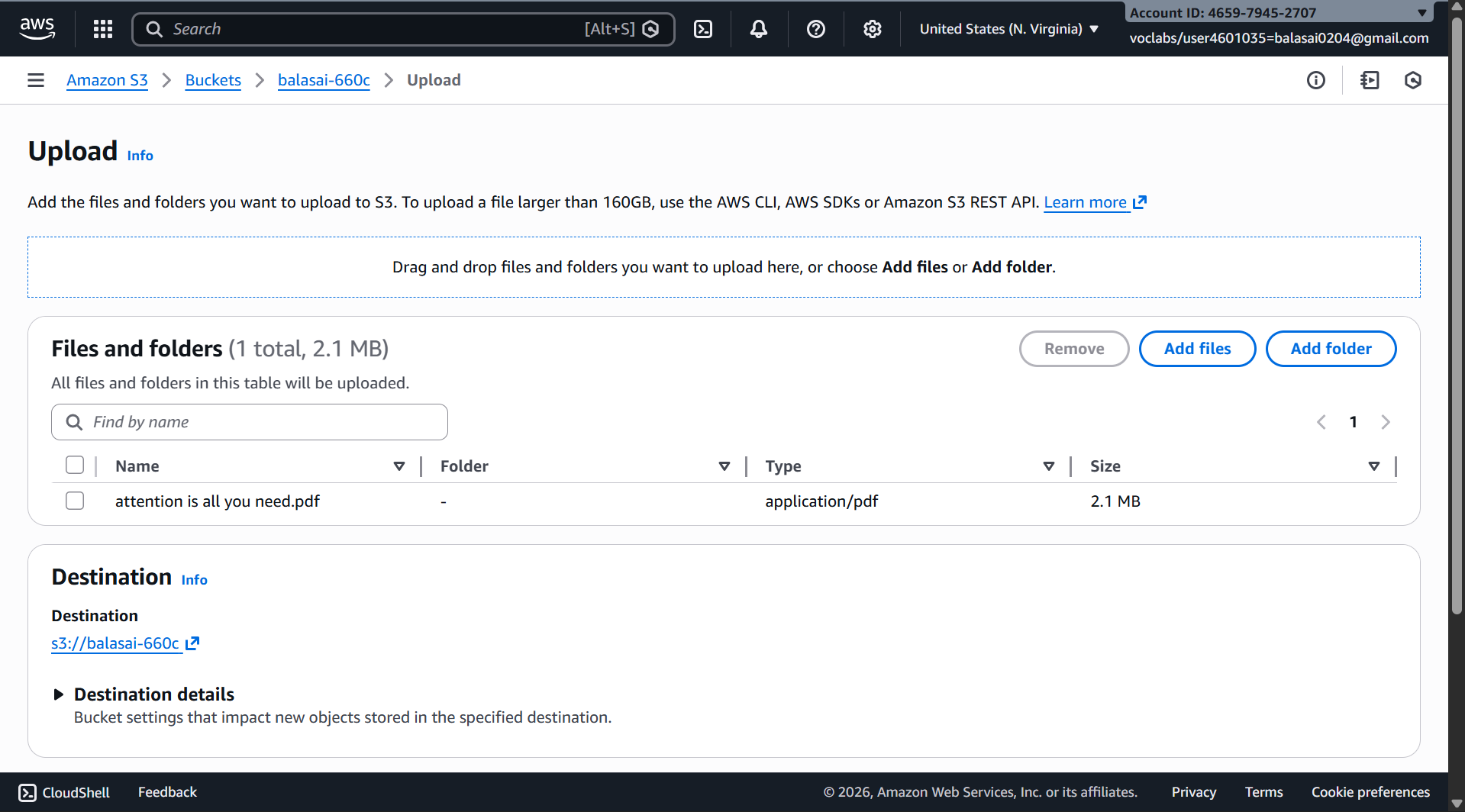
**🔸 Output Scenario (What You Observe)**

**Step A: Upload Object**

* Upload file: report.pdf

📌 Output:

* Version ID is assigned automatically

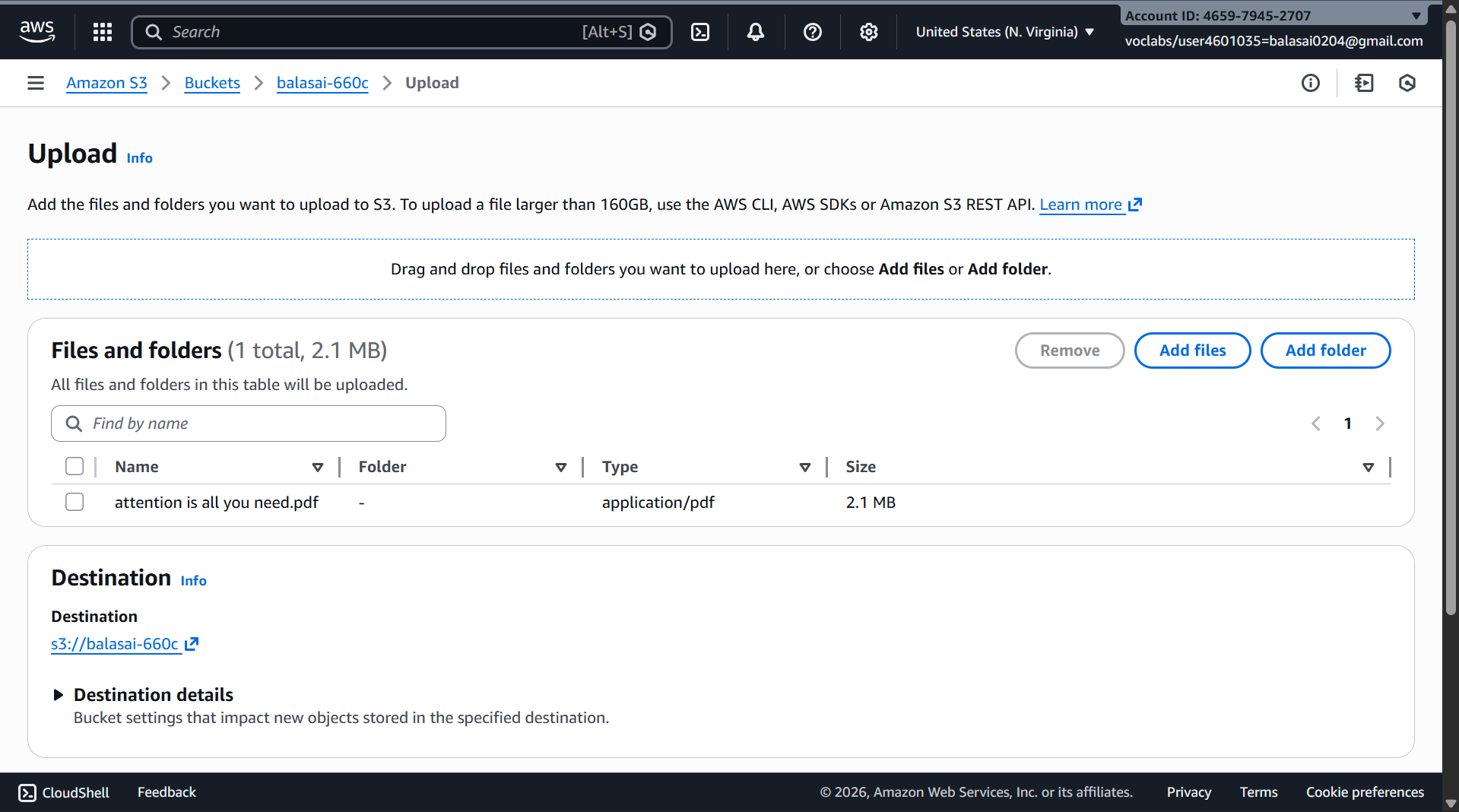


**Step B: Modify and Re-upload Same Object**

* **Upload report.pdf again (same name)**

**📌** Output:

* Old version is preserved
* New version becomes current

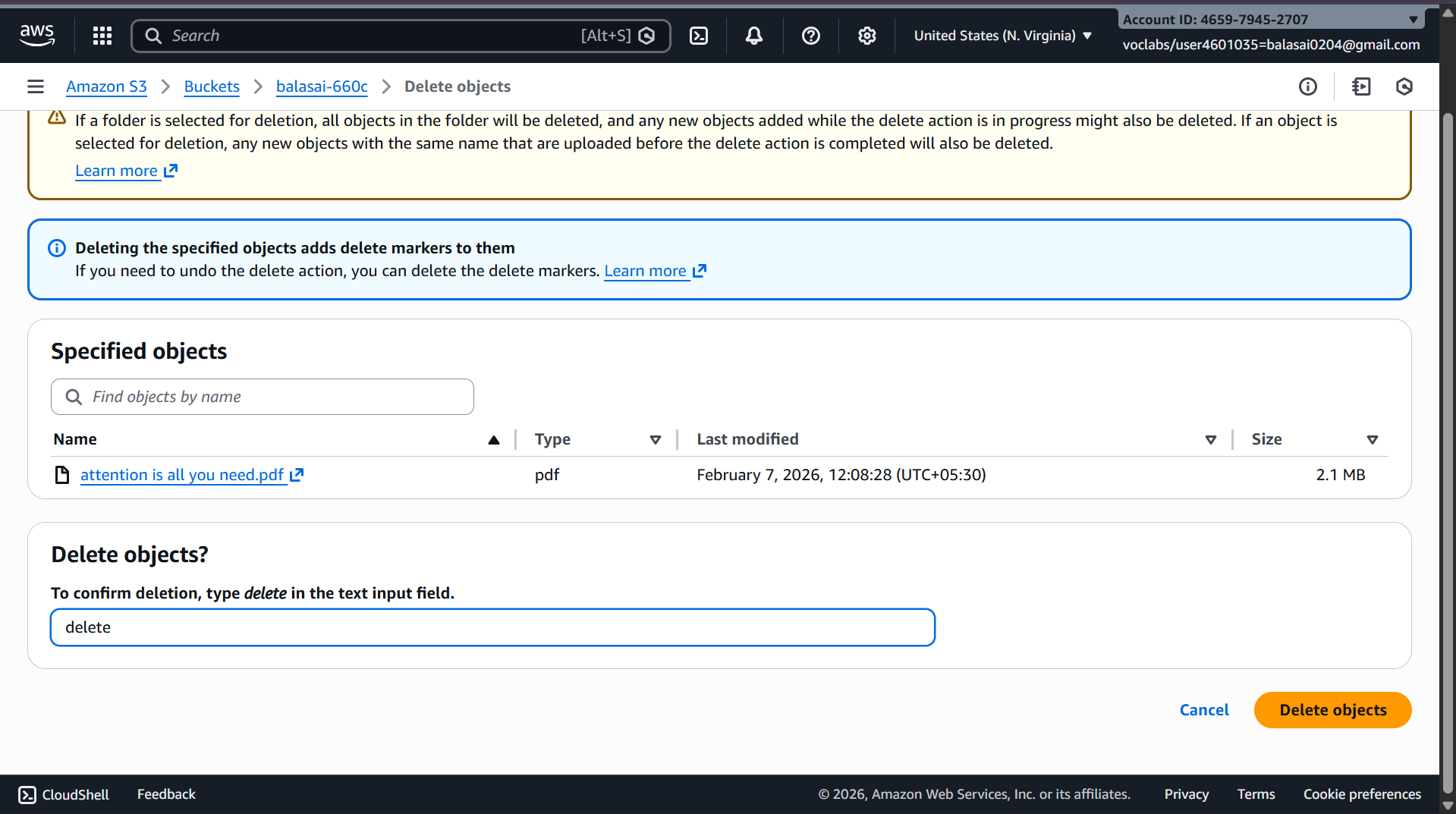


**Step C: Delete Object**

* Click Delete on report.pdf

**📌 Output:**

* Object appears deleted
* A Delete Marker is created
* Older versions still exist and can be restored



**2️⃣ Cross-Region Replication (CRR)**

**🔸 Objective**

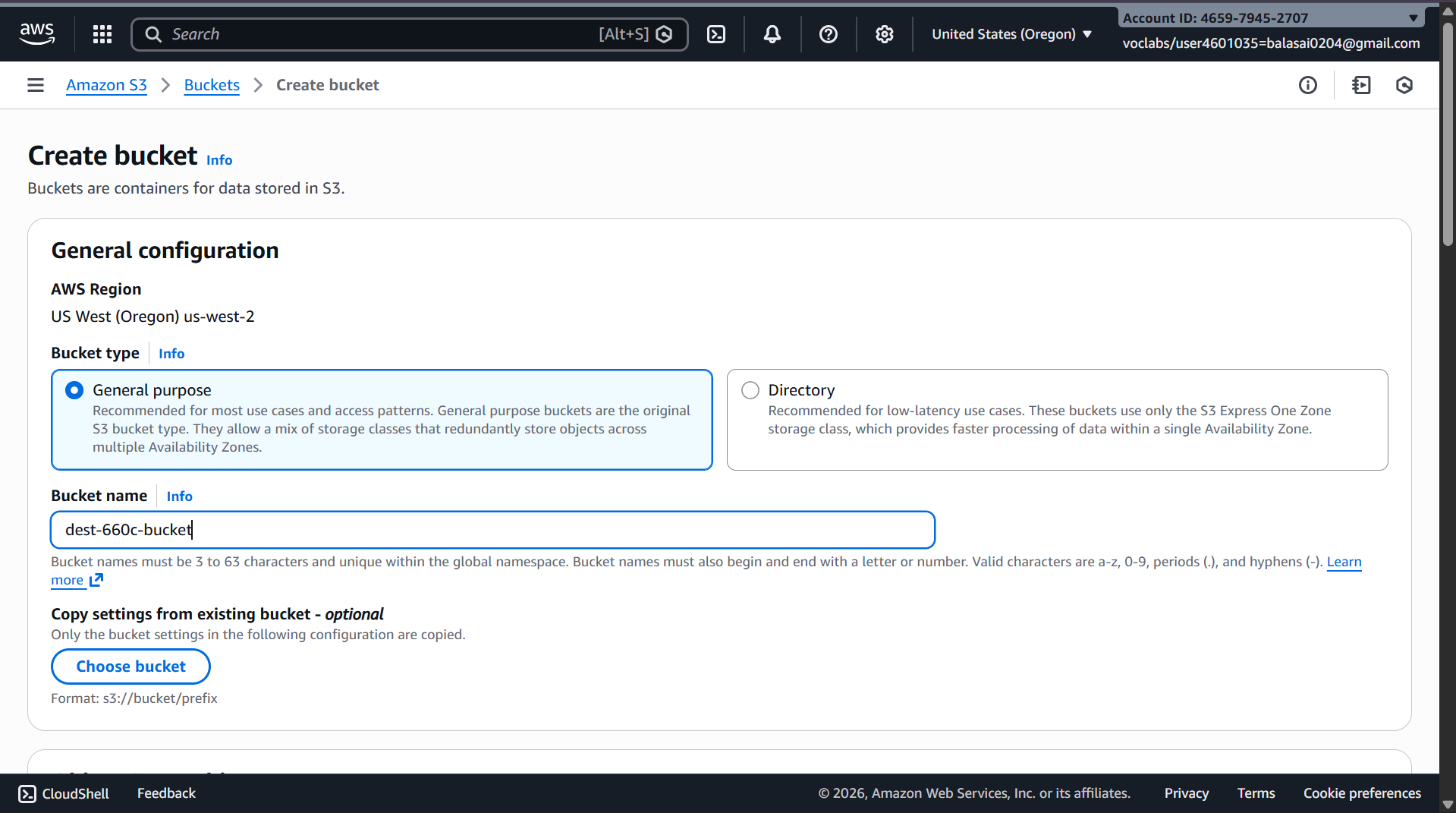
**To replicate objects from one bucket to another in a different region using AWS-managed permissions.**

**🔸 Prerequisites (Very Important)**

**✔** Versioning must be enabled on both buckets  
✔ Source and destination buckets must be in different regions  
✔ AWS Academy automatically handles replication permissions

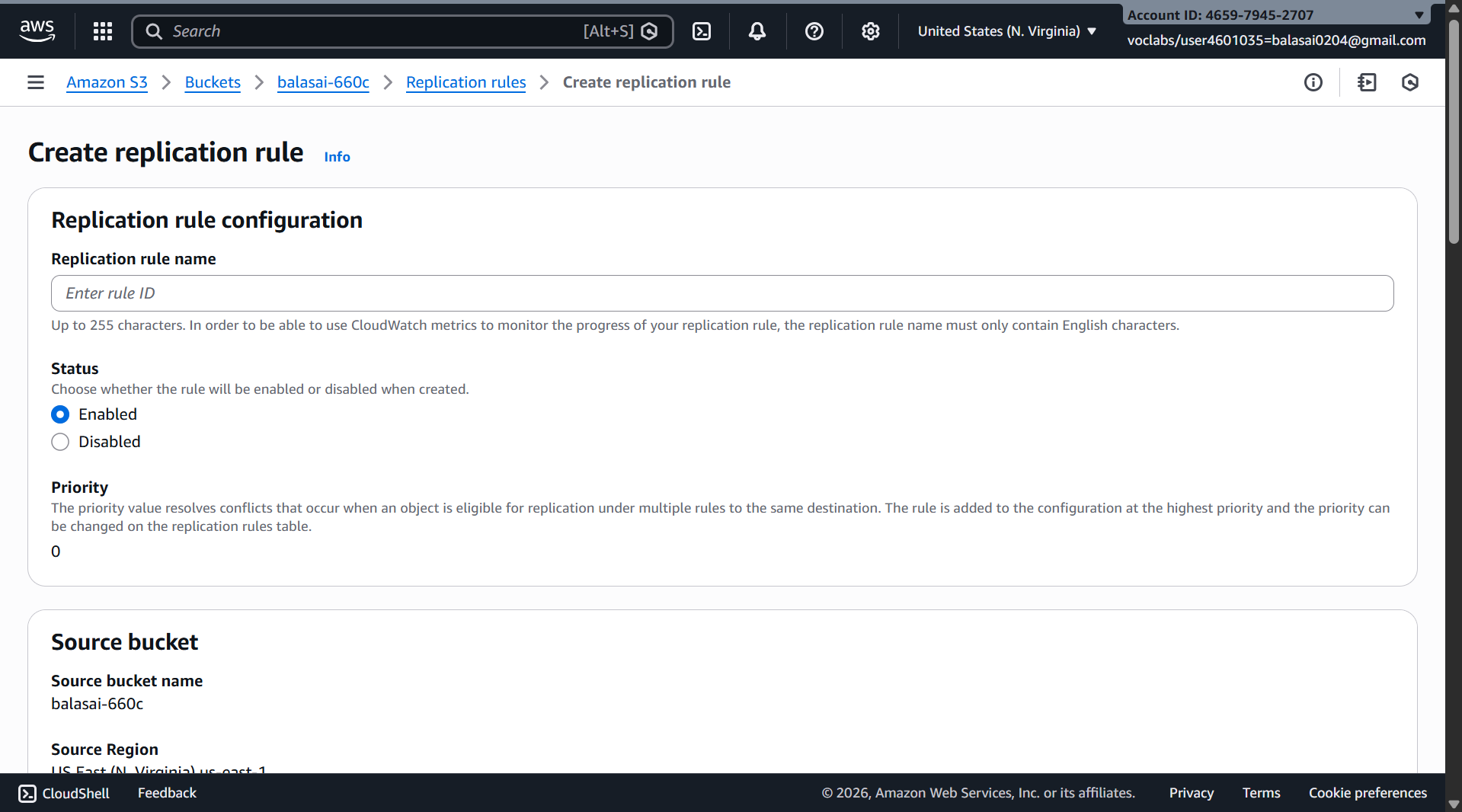
**Step 1️⃣ Create Destination Bucket**

1. Go to S3 → Create bucket
2. Bucket name: dest-crr-bucket
3. Select different region
4. Complete bucket creation
5. Enable Versioning (same steps as above)



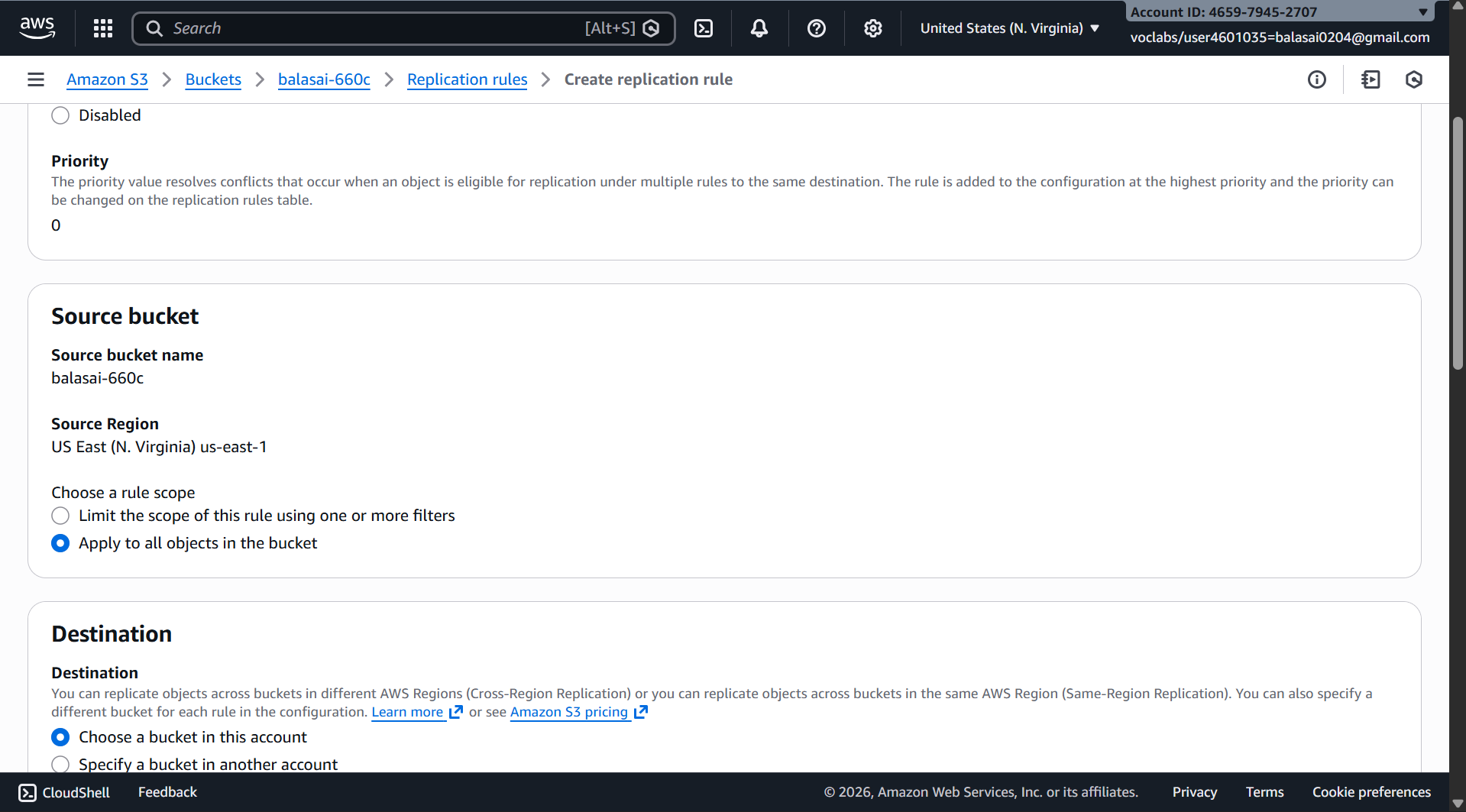
**Step 2️⃣ Create Replication Rule (Without IAM Selection)**

1. Open Source bucket
2. Go to Management tab
3. Click Replication rules
4. Click Create replication rule



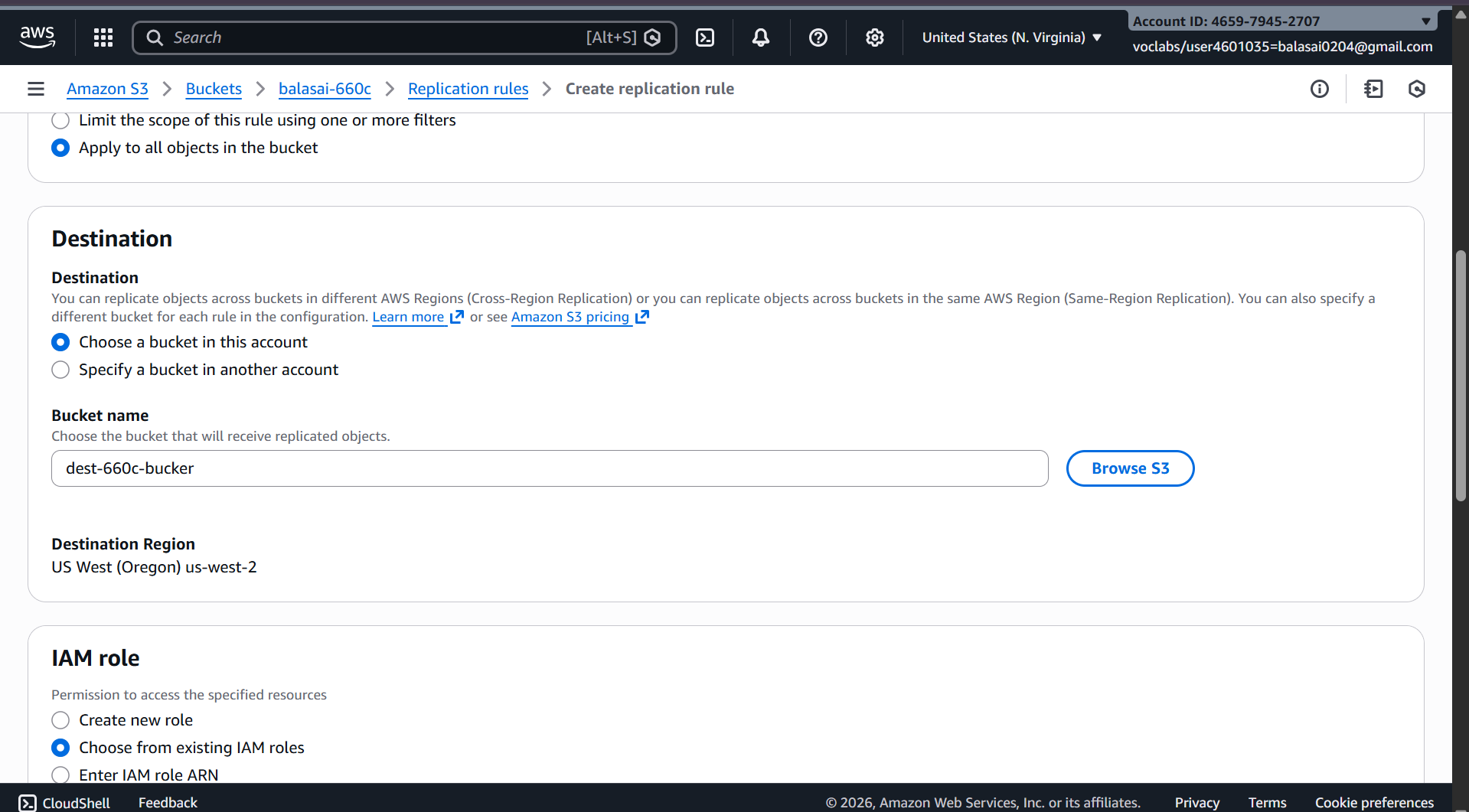
**Step 3️⃣ Configure Replication Rule**

* Rule name: academy-crr-rule
* Status: Enabled
* Scope: Apply to all objects



**Step 4️⃣ Choose Destination Bucket**

* Select Another bucket
* Choose Destination bucket
* Confirm different region



**📌 AWS Academy Note:  
You will see an option like:**

***"AWS will create and manage the required permissions automatically"***

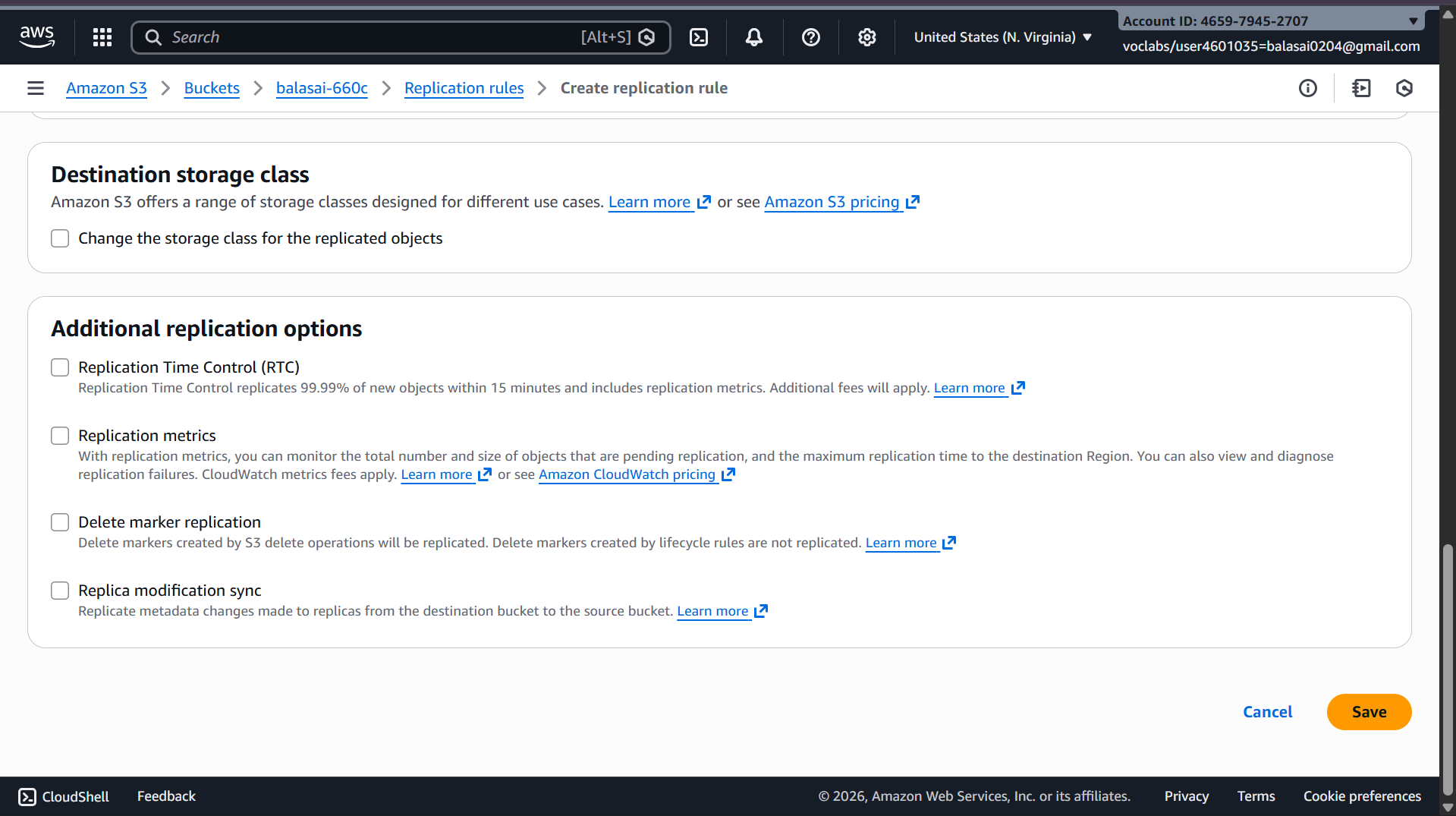
**✔ Select Use AWS-managed role / default role**

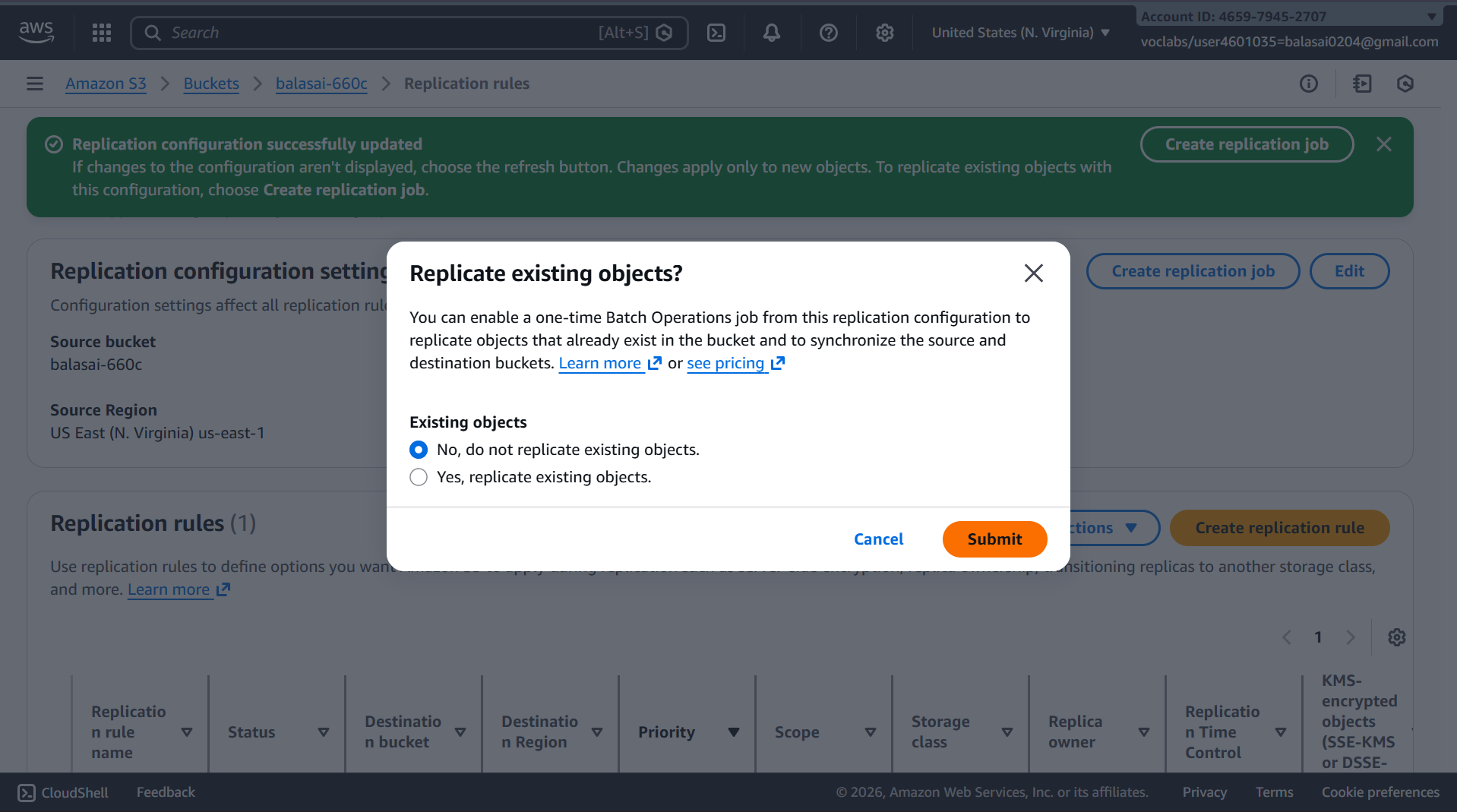
**Step 5️⃣ Replication Options**

* **Replicate:**
  + ✔ New objects
  + ❌ Existing objects (optional, depends on Academy permissions)
* **Encryption: Leave default**

**Click Save**

**✔ Replication rule is now active**

****



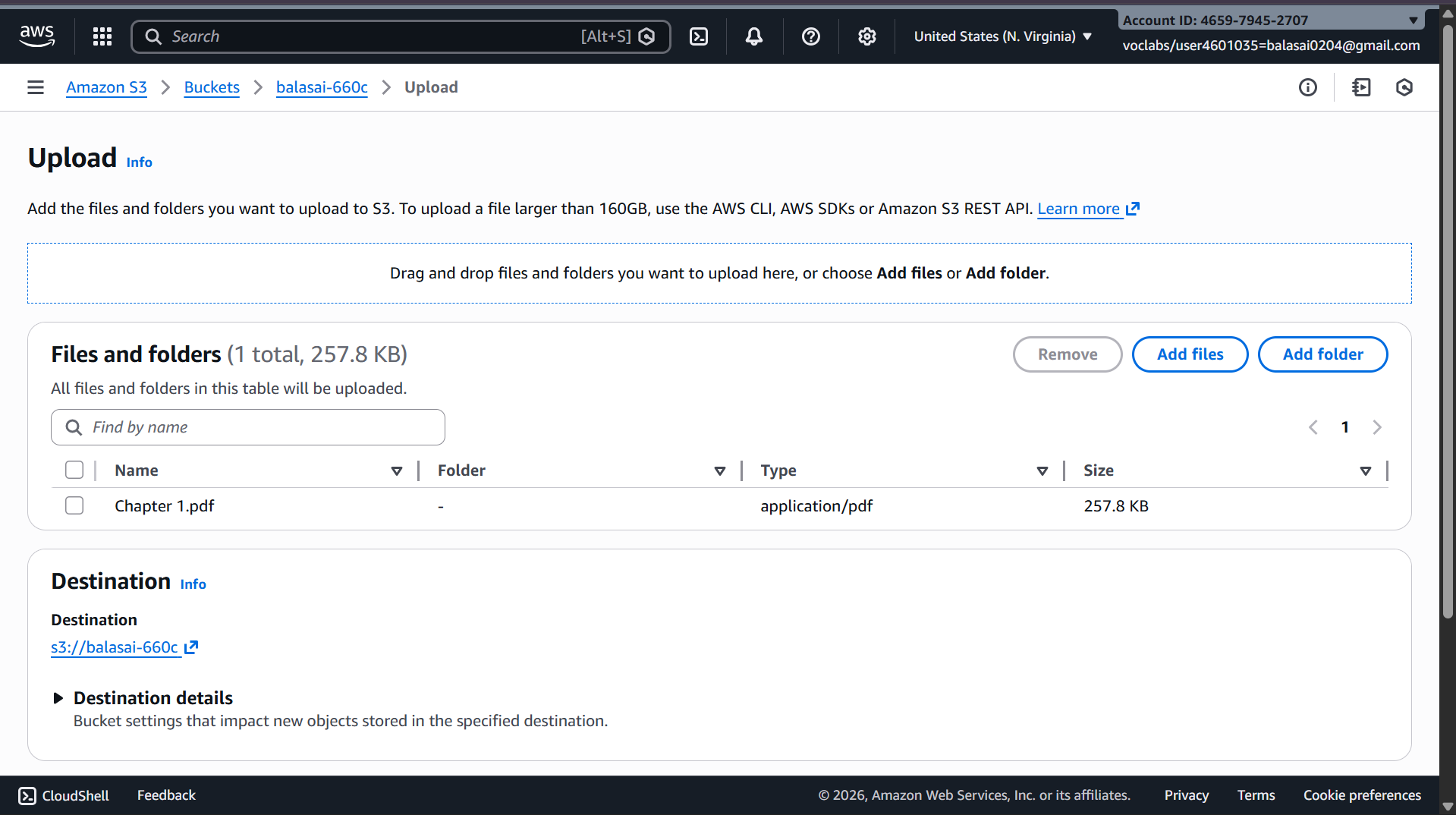
**3️⃣ CRR Output Scenario (What You Observe)**

**🔸** Step A: Upload New Object to Source Bucket

* Upload file: image.png

📌 Output:

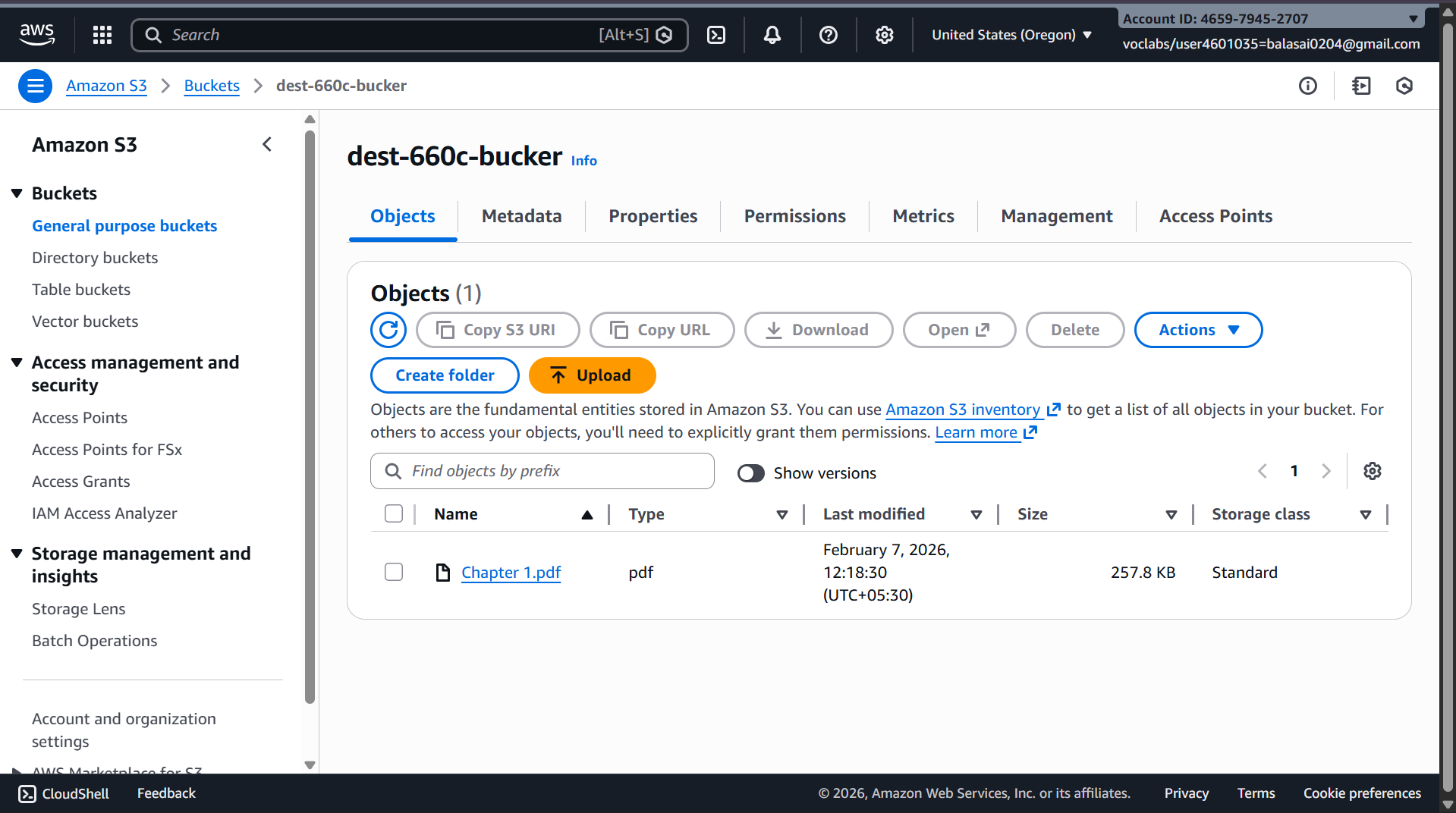
* Object is uploaded normally



**🔸 Step B: Verify Destination Bucket**

* Open destination bucket
* Object image.png appears automatically
* Version ID is present

**✔ Replication successful**

****

* 1. **Static Website Hosting in S3**

**(Using Bucket-Level & Object-Level Permissions – ACL Method)**

**🔹 Objective**

**To host a static website in Amazon S3 using index.html and error.html, and allow public access using ACL-based bucket and object permissions.**

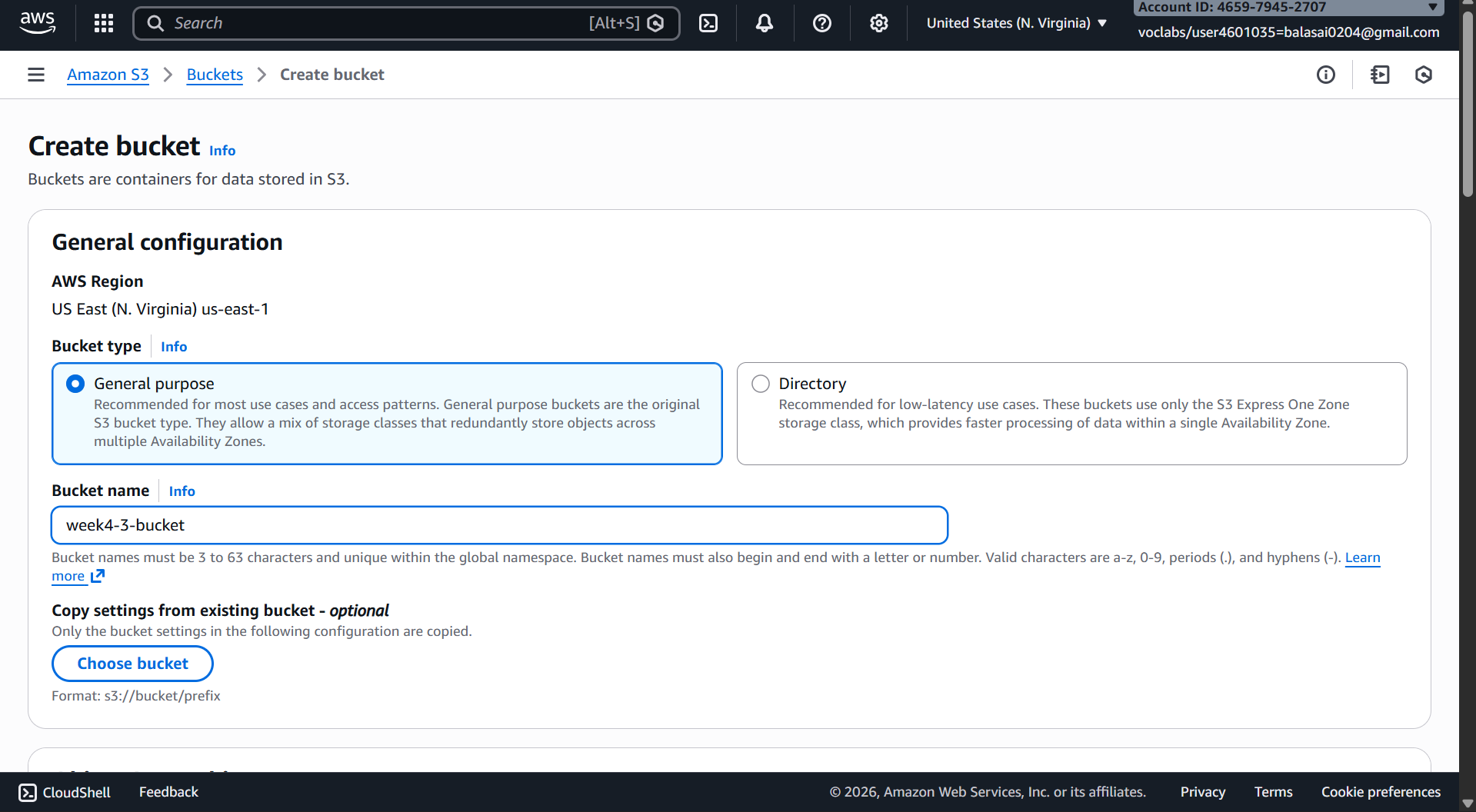
**🔹 Prerequisites**

* AWS / AWS Academy account
* Two files:
  + index.html
  + error.html

**1️⃣ Create an S3 Bucket**

1. Login to AWS Management Console
2. Go to Services → S3
3. Click Create bucket
4. **Enter:**
   * Bucket name: my-static-site-bucket-123
   * Region: Nearest region
5. **Under Block Public Access:**
   * ❌ Uncheck Block all public access
   * ✔ Acknowledge the warning
6. **Click Create bucket**

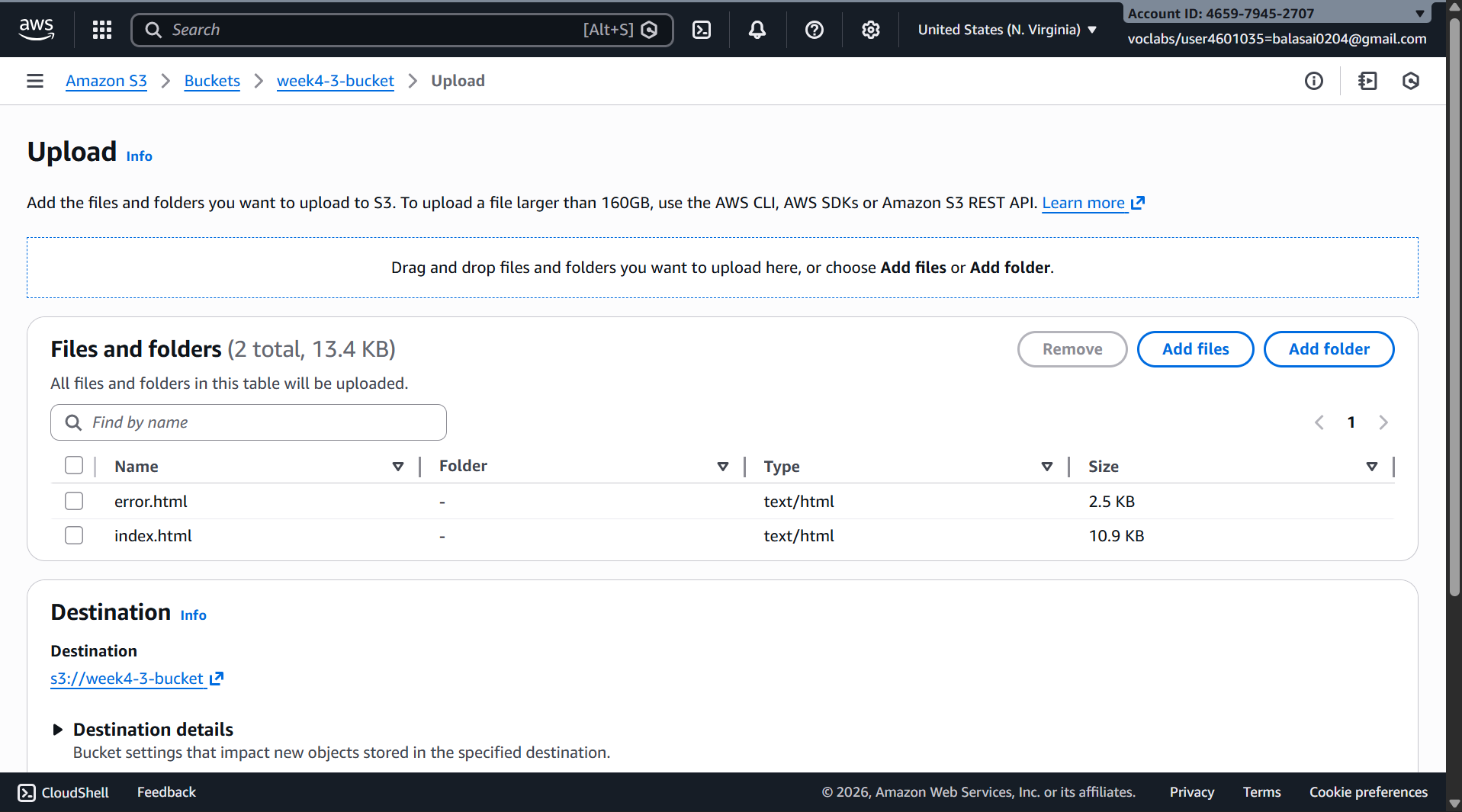
**✔ Bucket created successfully**

****

**2️⃣ Upload Website Files**

1. Open the bucket
2. Go to Objects tab
3. Click Upload
4. Click Add files
5. Select:
   * index.html
   * error.html
6. Click Upload

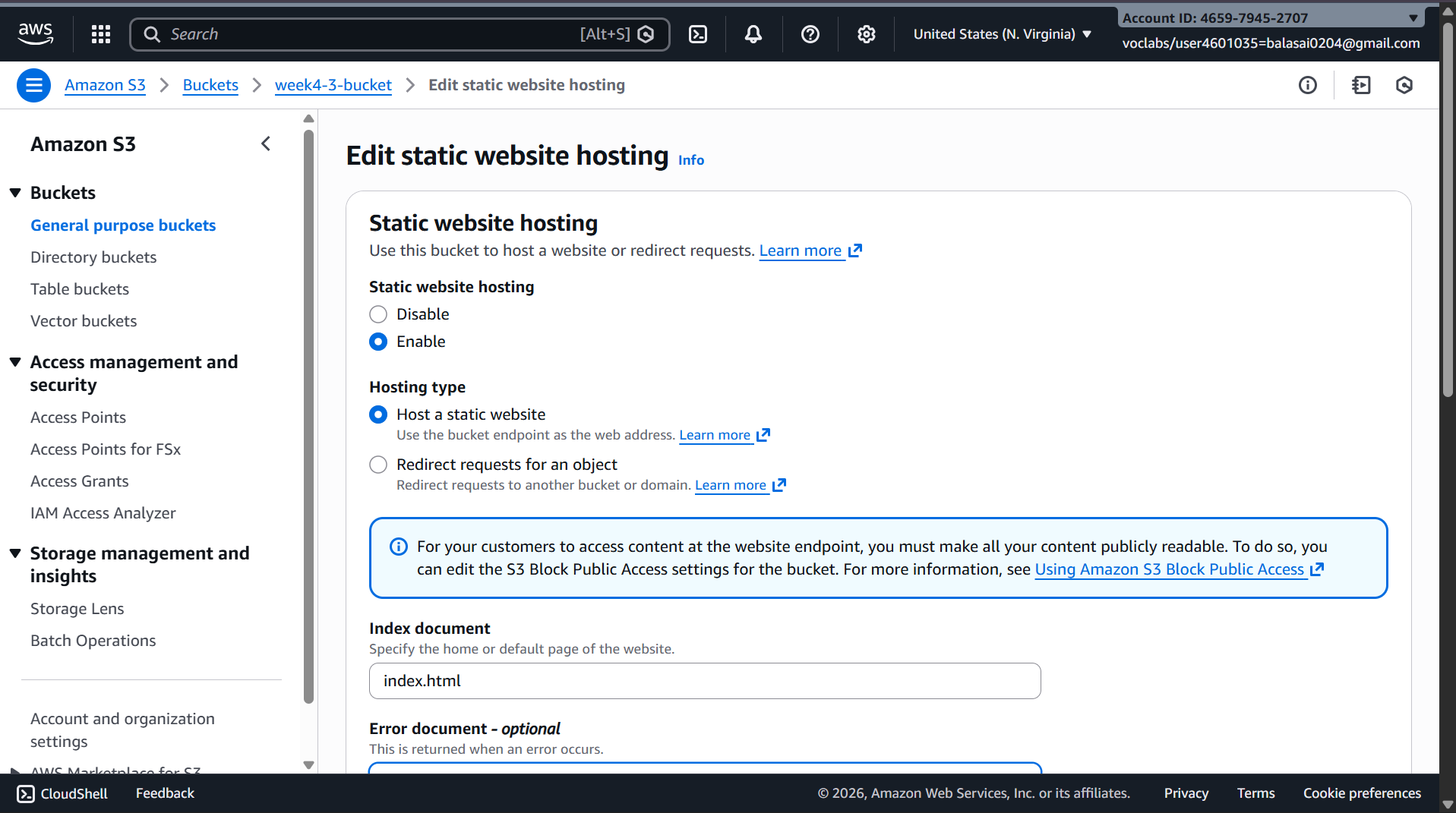
**✔ Files uploaded**

****

**3️⃣ Enable Static Website Hosting**

1. Go to Properties tab
2. Scroll to Static website hosting
3. Click Edit
4. Select Enable
5. Choose Host a static website
6. Enter:
   * Index document: index.html
   * Error document: error.html
7. **Click Save changes**

**✔ Static website hosting enabled**

****

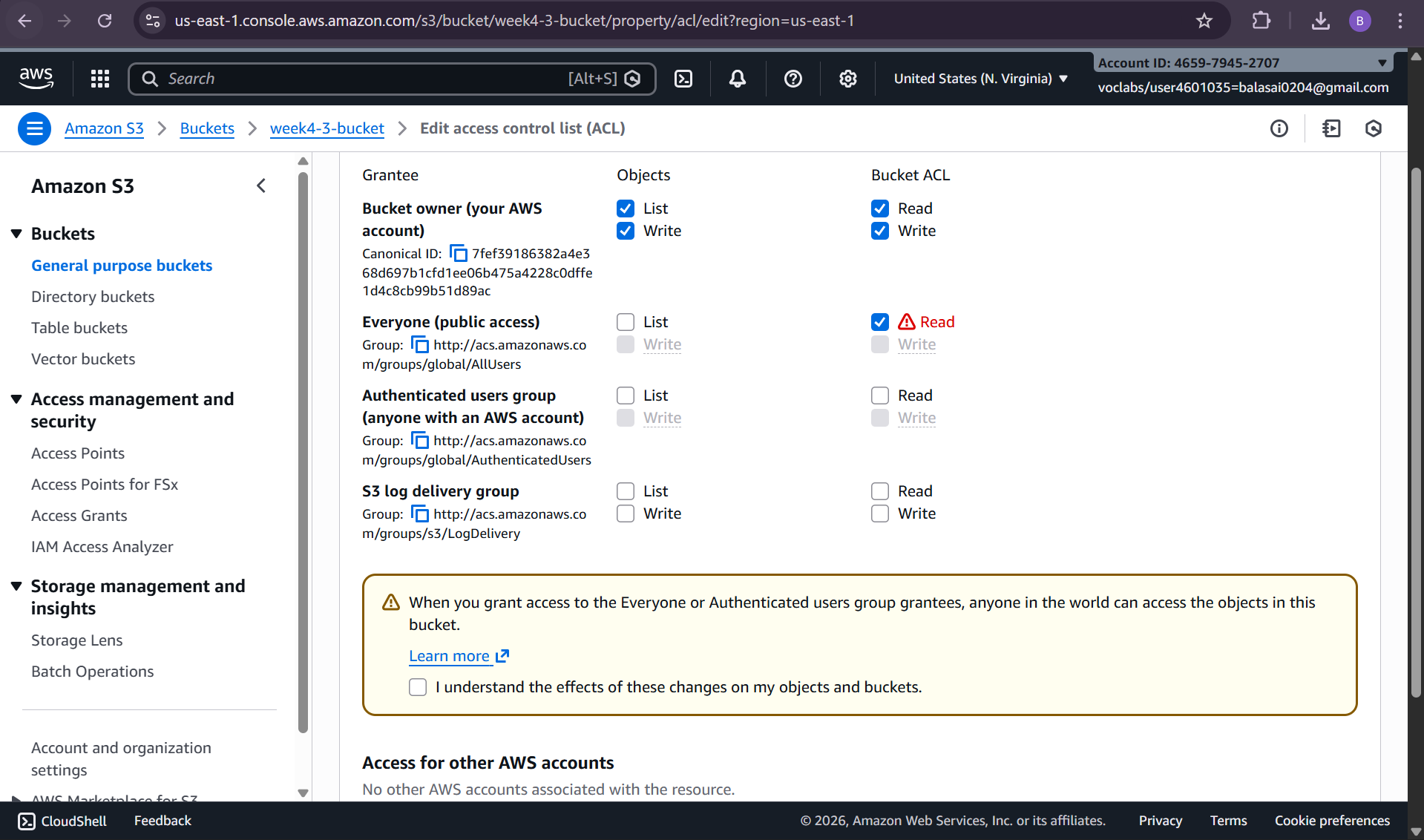
**4️⃣ Enable Bucket-Level Public Access (ACL)**

**🔹 Purpose**

**Allows public users to access objects inside the bucket.**

1. Go to Permissions tab
2. Scroll to Access Control List (ACL)
3. Click Edit
4. Under Public access:
   * Everyone (public access) → ✔ Read
5. Click Save changes

✔ Bucket-level public read access enabled



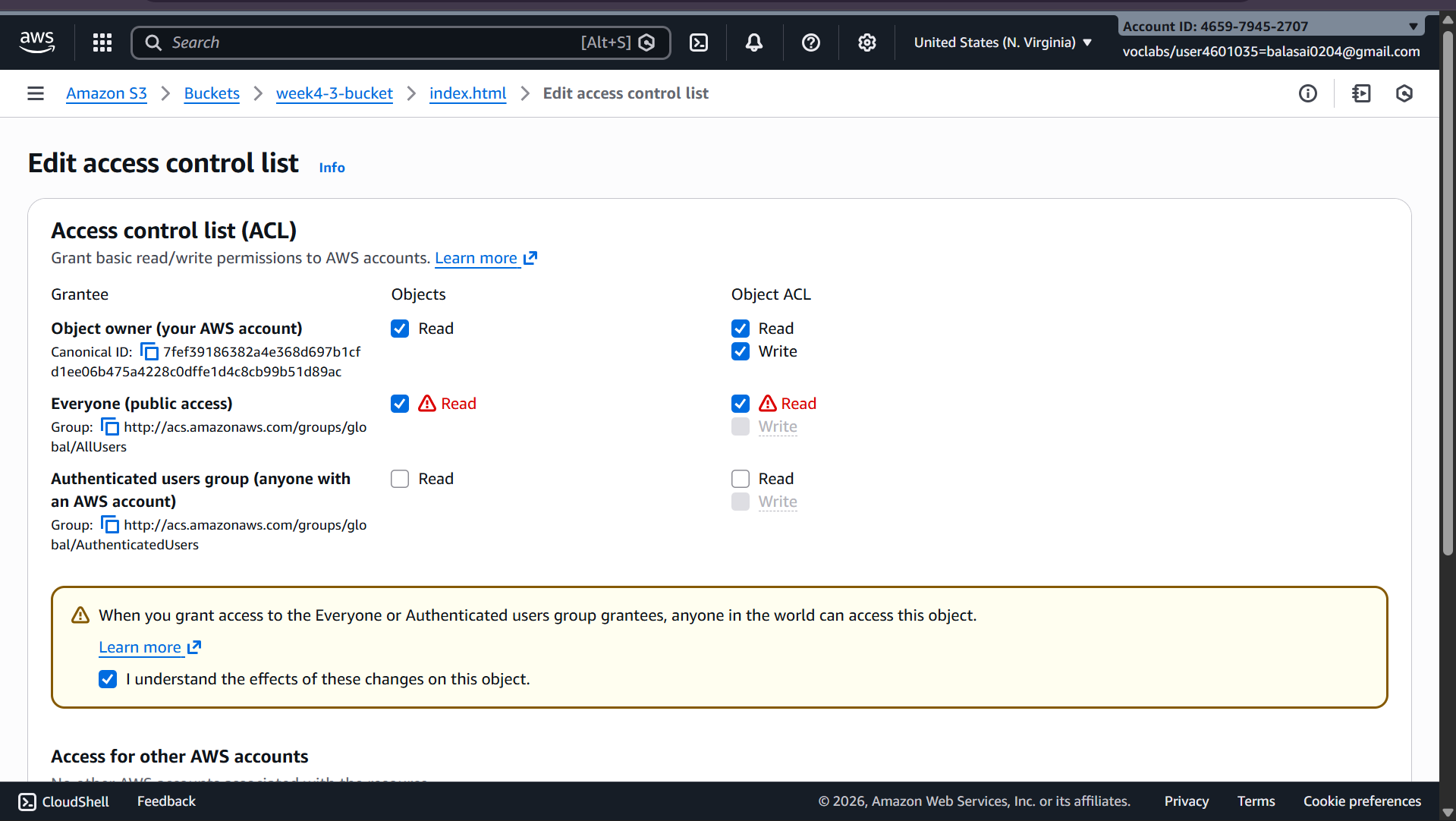
**5️⃣ Enable Object-Level Public Access (ACL)**

**Steps (For Each File)**

1. Go to Objects tab
2. Click on index.html
3. Go to Permissions
4. Scroll to Access Control List (ACL)
5. Click Edit
6. Under Public access:
   * Everyone (public access) → ✔ Read
7. Save changes

**🔁 Repeat the same steps for error.html**

**✔ Object-level public access enabled**

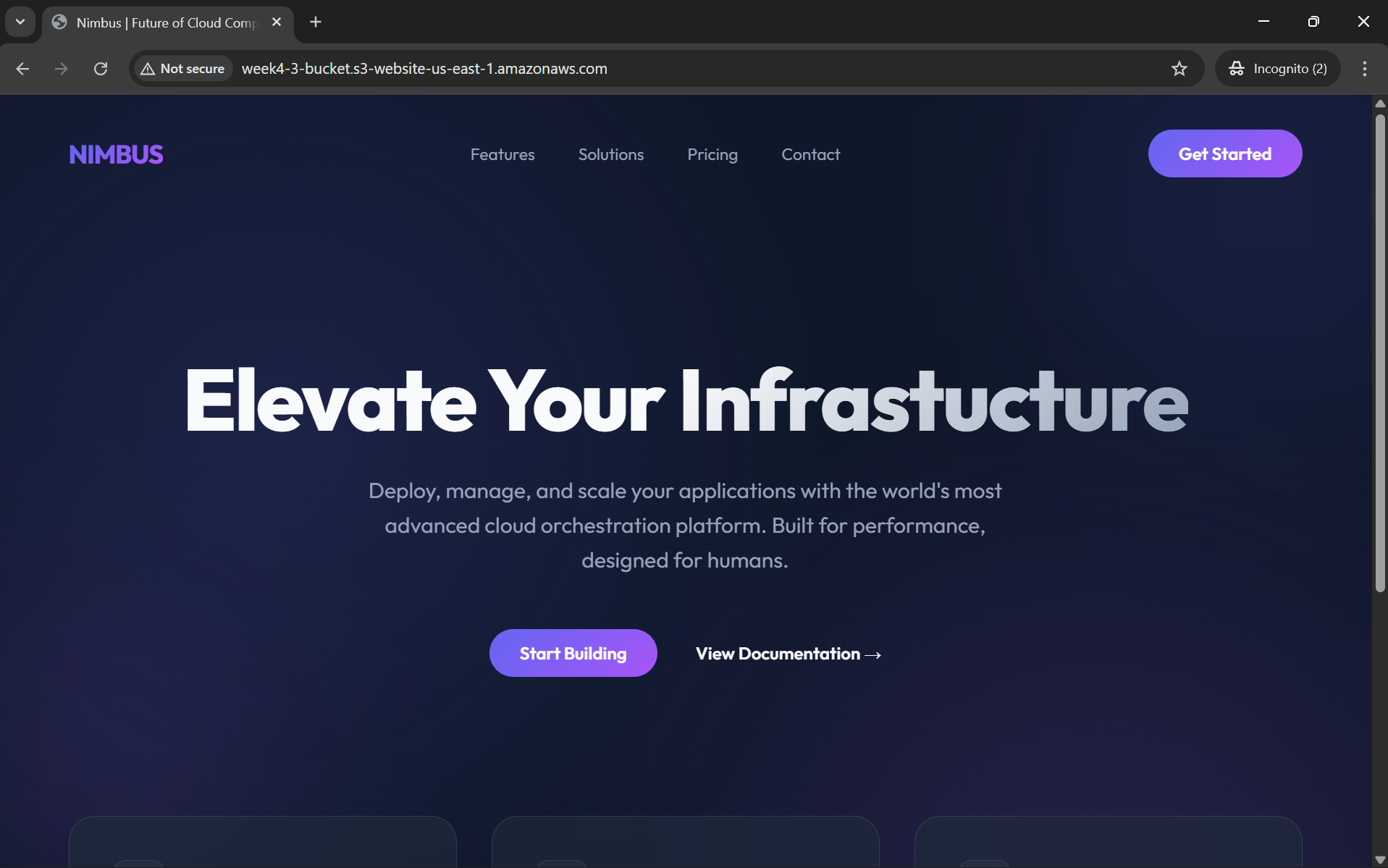
****

****

**6️⃣ Access the Static Website**

1. Go to Properties
2. Scroll to Static website hosting
3. Copy the Bucket website endpoint
4. http://my-static-site-bucket-123.s3-website-region.amazonaws.com
5. Paste into browser

✔ index.html page loads successfully



**7️⃣ Error Page Output Scenario**

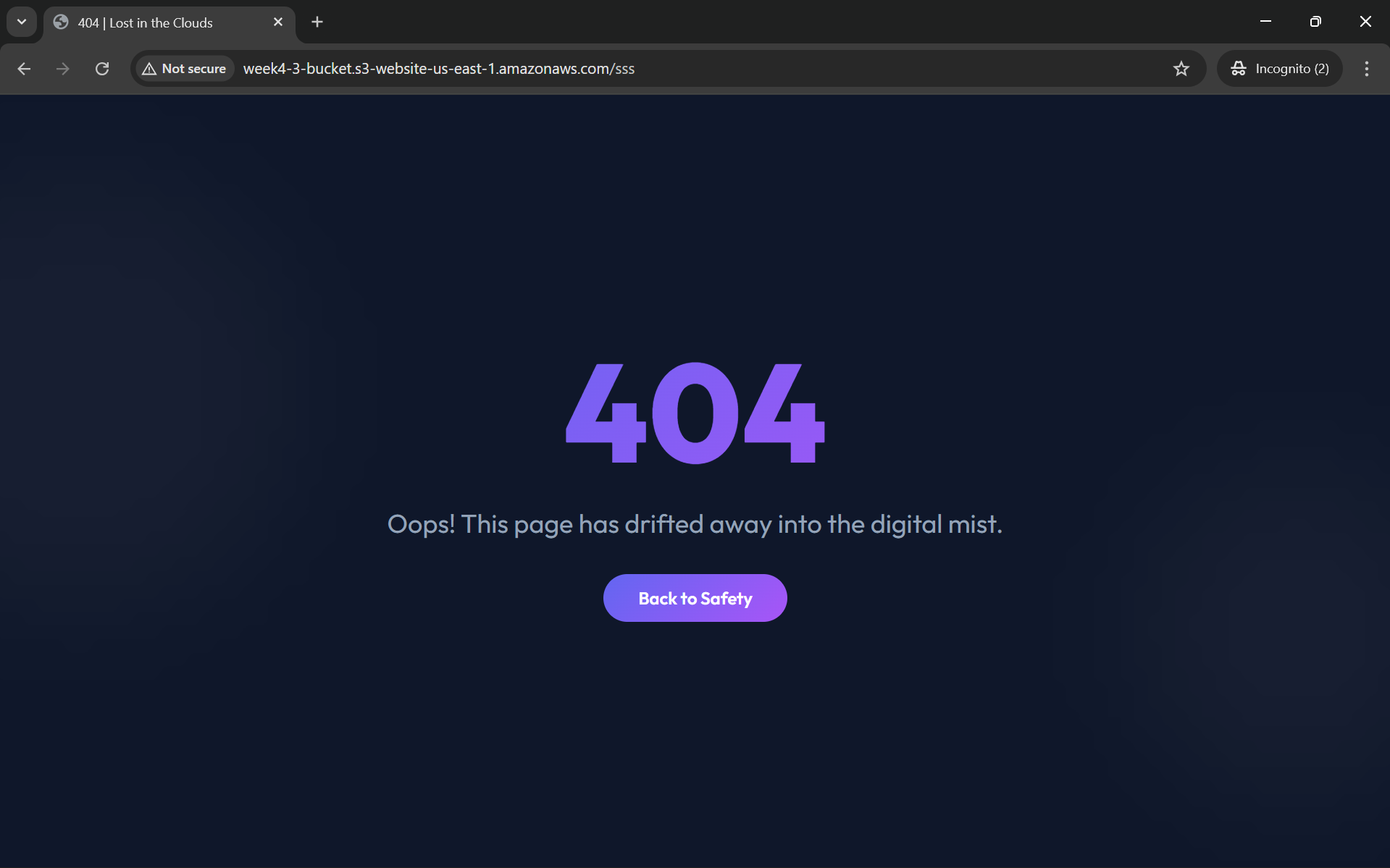
**🔹** Test Error Page

* Open an invalid URL:
* http://bucket-name.s3-website-region.amazonaws.com/invalid.html

📌 Output:

* error.html page is displayed

✔ Error page working correctly



**Scenario-Based Questions**

**1. A developer uploads an object to an S3 bucket but cannot access it from the browser, even though the bucket exists. What permission-related issue could be the cause?**

**Answer:**  
The object likely does not have **public read permission**. By default, S3 objects are private. Even if the bucket exists, the object must either:

* Have a bucket policy allowing public access, or
* Have an ACL granting public-read, or
* Be accessed via authenticated IAM permissions

Without proper permissions, the browser returns **403 Forbidden**.

**2. An S3 bucket policy allows public read access, but a specific object inside the bucket is still not accessible. Why might this happen?**

**Answer:**  
Possible reasons:

* The object ACL explicitly denies access.
* **S3 Block Public Access** settings are enabled.
* There is an explicit Deny statement in the bucket policy.
* The object is encrypted with KMS and lacks KMS permissions.

Explicit deny always overrides allow.

**3. A bucket has no bucket policy, but individual objects have public ACLs. Will users be able to access those objects? What determines the final access decision?**

**Answer:**  
Yes, users can access those objects **if S3 Block Public Access is not enabled**.

Final access decision depends on:

1. IAM policies
2. Bucket policies
3. Object ACLs
4. S3 Block Public Access settings

If Block Public Access is enabled, public ACLs are ignored.

**4. A company wants to ensure no public access to any object in an S3 bucket, even if someone mistakenly applies a public ACL. What S3 feature should be enabled?**

**Answer:**  
Enable **S3 Block Public Access (BPA)** at the bucket or account level.

This overrides:

* Public ACLs
* Public bucket policies

It ensures no accidental public exposure.

**5. You attempt to create an S3 bucket named my-company-data but receive a naming error. What S3 bucket naming rules might have been violated?**

**Answer:**  
Possible violations:

* Bucket name must be globally unique.
* Must be between 3–63 characters.
* Only lowercase letters, numbers, dots (.), and hyphens (-).
* Cannot start or end with a hyphen or dot.
* Cannot look like an IP address (e.g., 192.168.1.1).

Most commonly: the name is already taken globally.

**6. Two different AWS accounts try to create a bucket named project-backup-2026. One succeeds, the other fails. Why?**

**Answer:**  
S3 bucket names are **globally unique across all AWS accounts**.

Once one account creates project-backup-2026, no other account in any region can use that name.

**7. You enabled static website hosting on an S3 bucket and uploaded files, but accessing the website URL returns 403 Forbidden. What is the most likely cause?**

**Answer:**  
The bucket or objects are not publicly accessible.

Static website hosting requires:

* Public read access on objects
* Bucket policy allowing public read
* Block Public Access disabled (or configured properly)

Without these, the website endpoint returns **403 Forbidden**.

**8. Static website hosting is enabled, but accessing the root URL shows an error saying index.html not found. What is the actual issue?**

**Answer:**  
The **index document is either missing or misnamed**.

S3 is configured to look for index.html (or whatever was specified). If the file does not exist at the root level, the site fails.

**9. What HTTP error is returned when the index document is missing in S3 static website hosting, and why does this happen?**

**Answer:**  
**HTTP 404 Not Found**

This happens because S3 looks for the configured index document and cannot find it in the bucket.

**10. A bucket policy allows public access, but static website hosting still doesn’t work. What object-level setting must also be verified?**

**Answer:**  
Verify that:

* The object itself has correct permissions.
* It is not encrypted with KMS (without public decrypt permission).
* Object ownership settings (ACL disabled / Bucket Owner Enforced mode).

Object-level encryption and ACL settings can still block access.

**11. You accidentally overwrite an important object in S3. Versioning was enabled earlier. How can the original file be recovered?**

**Answer:**  
When versioning is enabled:

1. Go to the object in S3.
2. Enable **Show Versions**.
3. Locate the previous version.
4. Download or restore it.

Each overwrite creates a new version — old versions are preserved.

**12. A user deletes an object from a versioned S3 bucket. Is the data permanently removed? Explain what actually happens.**

**Answer:**  
No, it is not permanently removed.

S3 adds a **Delete Marker** instead.

* The delete marker becomes the current version.
* Older versions still exist.
* You can delete the delete marker to restore access.

Permanent deletion happens only if all versions are deleted.

**13. You enabled Cross-Region Replication (CRR) on a bucket, but objects are not replicating. What configuration might be missing?**

**Answer:**  
Possible missing configurations:

* Versioning not enabled on both source and destination buckets.
* IAM replication role not configured.
* Missing replication rule.
* No proper bucket policy on destination bucket.

CRR requires versioning enabled on both buckets.

**14. After enabling CRR, only newly uploaded objects are replicated, but old objects are not. Why?**

**Answer:**  
CRR only replicates **new objects created after replication is enabled**.

Existing objects are not automatically replicated.

To replicate old objects, you must use:

* S3 Batch Replication
* Or manually re-upload/copy objects

**15. A company wants to replicate S3 data to another region for disaster recovery and compliance. What are the key prerequisites for enabling CRR?**

**Answer:**

Key prerequisites:

1. Versioning enabled on both buckets.
2. Destination bucket in a different AWS region.
3. IAM role with replication permissions.
4. Proper replication rule configuration.
5. Bucket policy allowing replication from source.

These ensure secure and automated cross-region replication.