

AWS S3

Why use Amazon S3? — Full list (every point included)

When comparing **EBS**, **EFS**, and **S3**, here's a complete, point-by-point explanation of why we use **S3** for object storage.

Key reasons

1. Limited storage in EBS.

EBS volumes are fixed-size — you pay for the full volume even if you don't use it.

2. EFS is expensive.

Example (approx):

- EFS → 1 GB ≈ ₹30
- S3 → 1 GB ≈ ₹2

3. Hard to manage / track multiple EBS volumes.

When you store many files across multiple EBS volumes it becomes hard to track. S3 centralizes objects in a bucket and is easier to manage.

4. EFS doesn't provide direct HTTP URLs like S3.

S3 gives object endpoints / URLs you can open in a browser.

5. Different primary use-cases.

- EFS → application data (shared filesystem, source code)
- S3 → object storage (files, images, backups, media, static websites)

6. Unlimited storage.

S3 scales automatically — you pay only for what you use.

7. Common/shared storage.

Multiple applications or users can access the same bucket.

8. Redundant storage across AZs.

Objects are stored redundantly across 3 or more Availability Zones.

9. Very high availability.

Availability is very high (e.g., ~99.999% or better depending on class).

10. Very high durability.

"11 nines" durability — 99.99999999% — designed to prevent data corruption.

11. Data in S3 is encrypted by default.

This helps keep data secure at rest.

12. Fine-grained permissions.

Unlike EBS folders, S3 allows policies at object-level (per file) for precise access control.

13. Protection against accidental deletes.

Use Versioning and Object Lock (or bucket policies) to prevent accidental deletion.

Short summary

- Unlimited storage
 - Common shared storage
 - 11 nines → durability
 - Very high availability (~99.999%)
 - Low cost — pay only for what you use
-

S3 object basics

- **Bucket name must be globally unique.**
- S3 stores **objects**. Each object contains:
 1. File
 2. Metadata
 3. Key → the "path" of the file inside the bucket

- **S3 does not have real folders.**

Folders are a UI concept created from object keys.

Bucket endpoint example (shows folder-like path)

```
http://<bucket_name>.s3.<region>.amazonaws.com/images/abc.jpg
```

/images/abc.jpg looks like a folder path but it is actually part of the object key.

Default access & public access

- **By default, S3 buckets are private.**
 - To access files publicly you must set bucket or object permissions.
 - You can make the **bucket public** and then allow public access only to selected objects via policies.
-

ACL disabled vs ACL enabled

1. ACL disabled (recommended)

- Use bucket policies / IAM for fine-grained control.
- Everything is private by default.
- Allows to define permission at deep level.
- You can allow or deny specific actions (upload, read, delete).

2. ACL enabled

- Simple Read/Write permissions via ACLs.
 - Less flexible than policies. Good for simple use-cases.
-

Creating ACL Enabled Bucket (Practical)

1. Go to AWS Console → S3.

Write bucket name → select **ACL enabled**.

Create bucket Info

Buckets are containers for data stored in S3.

General configuration

AWS Region
US East (N. Virginia) us-east-1

Bucket type Info

General purpose
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

Director
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name Info
19112025acenabled

Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). [Learn more](#)

Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.

Choose bucket Info
Format: s3://bucket/prefix

Object Ownership Info
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Block Public Access settings for this bucket
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or AI. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

Block public access to buckets and objects granted through new access control lists (ACLs)
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

Block public access to buckets and objects granted through any access control lists (ACLs)
S3 will ignore all ACLs that grant public access to buckets and objects.

Block public access to buckets and objects granted through new public bucket or access point policies
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

Block public and cross-account access to buckets and objects through any public bucket or access point policies
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

I acknowledge that the current settings might result in this bucket and the objects within becoming public.

Bucket Versioning
Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Disable

Enable

2. Allow public access for the bucket if needed.

Block Public Access settings for this bucket
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or AI. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

Block public access to buckets and objects granted through new access control lists (ACLs)
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

Block public access to buckets and objects granted through any access control lists (ACLs)
S3 will ignore all ACLs that grant public access to buckets and objects.

Block public access to buckets and objects granted through new public bucket or access point policies
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

Block public and cross-account access to buckets and objects through any public bucket or access point policies
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

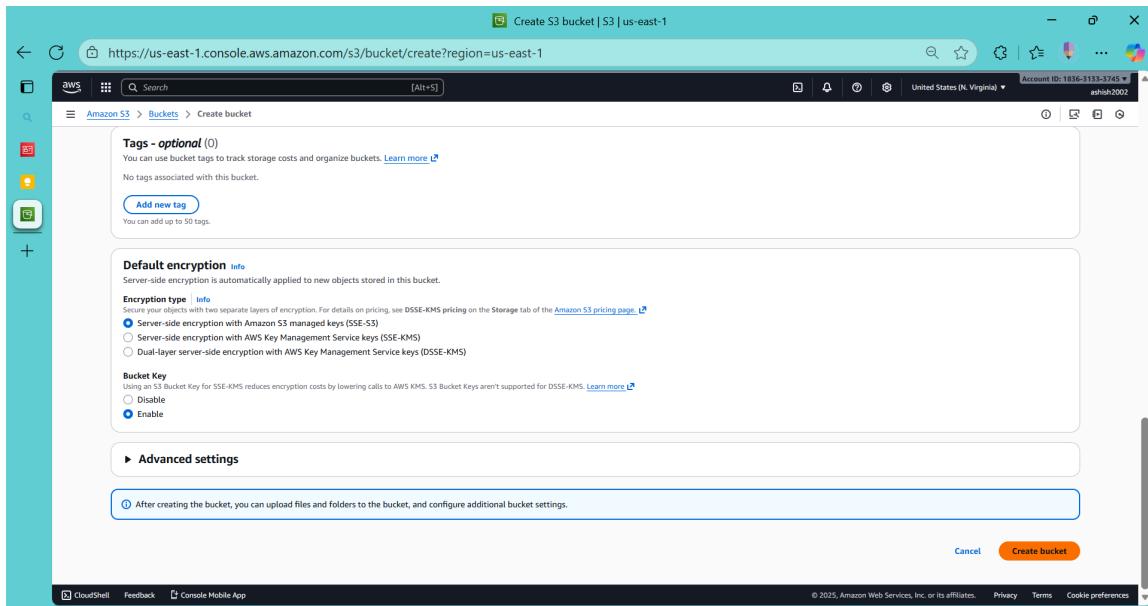
I acknowledge that the current settings might result in this bucket and the objects within becoming public.

Bucket Versioning
Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

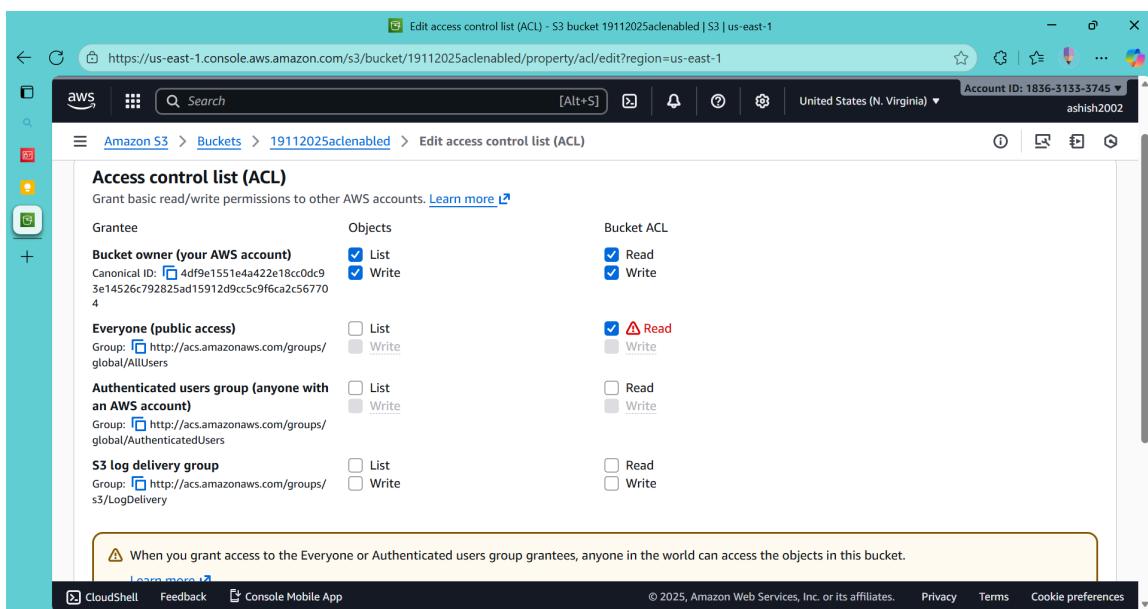
Disable

Enable

3. Create the bucket.



4. Give Read permission to the bucket if you want objects to be publicly readable.



5. Upload the image or file.

The screenshot shows the AWS S3 console interface. At the top, there's a header with the title "General purpose buckets (13)" and a "Info" link. Below the header are several buttons: "Copy ARN", "Empty", "Delete", and a prominent orange "Create bucket" button. A sub-header states "Buckets are containers for data stored in S3." There's a search bar labeled "Find buckets by name". Below the search bar is a table with three columns: "Name", "AWS Region", and "Creation date". A single row is visible, representing the bucket "19112025aclenabled" which was created in the "US East (N. Virginia) us-east-1" region on "November 19, 2025, 13:56:40 (UTC+05:30)".

Click **Upload** and upload the file.

This screenshot shows the "Objects" tab of the AWS S3 bucket "19112025aclenabled". The tab bar includes "Objects" (which is selected), "Metadata", "Properties", "Permissions", "Metrics", "Management", and "Access Points". The main area displays a message stating "No objects" and "You don't have any objects in this bucket.". At the bottom of the page, there is a blue "Upload" button.

6. Grant public read access for that uploaded object.

This screenshot shows the "Upload" dialog for the "19112025aclenabled" bucket. In the "Access control list (ACL)" section, the "Grant public-read access" option is selected. A warning message states: "Granting public-read access is not recommended. Anyone in the world will be able to access the specified objects." Below this, a checkbox is checked with the text "I understand the risk of granting public-read access to the specified objects." At the bottom right of the dialog are "Cancel" and "Upload" buttons.

7. Copy the object URL and paste it into a browser.

The screenshot shows the AWS S3 console with the path `Buckets > 19112025aclenabled > a.png`. The left sidebar lists various bucket types. The main panel displays the **Properties** tab for the object `a.png`. Key details shown include:

- S3 URI:** `s3://19112025aclenabled/a.png`
- Amazon Resource Name (ARN):** `arn:aws:s3:::19112025aclenabled/a.png`
- Entity tag (Etag):** `5a4e60a03218027e66012ef930103cbd`
- Object URL:** `https://19112025aclenabled.s3.us-east-1.amazonaws.com/a.png`
- Owner:** ashish2002 (with a long hex string)
- AWS Region:** US East (N. Virginia) us-east-1
- Last modified:** November 19, 2025, 14:01:13 (UTC+05:30)
- Size:** 696.3 KB
- Type:** png
- Key:** a.png

You will see the uploaded image.

The screenshot shows a terminal window with the command `ifconfig` run on an AWS Lambda environment. The output shows two interfaces:

- docker0:** flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
ether 02:42:bd:51:80:23 txqueuelen 0 (Ethernet)
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
- eth0:** flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 172.27.95.133 netmask 255.255.240.0 broadcast 172.27.95.255
inet6 fe80::215:5dff:fec5:58d3 prefixlen 64 scopeid 0x20<link>
ether 00:15:5d:c5:58:d3 txqueuelen 1000 (Ethernet)
RX packets 272 bytes 241018 (241.0 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 143 bytes 10686 (10.6 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

8. Revoke permission for everyone to test access — you will see **Access Denied** for that object.

Access control list (ACL)

Grant basic read/write permissions to AWS accounts. [Learn more](#)

Grantee	Objects	Object ACL
Object owner (your AWS account)	<input checked="" type="checkbox"/> Read	<input checked="" type="checkbox"/> Read <input checked="" type="checkbox"/> Write
Canonical ID: 4df9e1551e4a422e18cc0dc93e14526c792825ad15912d9cc5c9f6ca2c56704		
Everyone (public access)	<input type="checkbox"/> Read	<input type="checkbox"/> Read <input type="checkbox"/> Write
Group: http://acs.amazonaws.com/groups/global/AllUsers		
Authenticated users group (anyone with an AWS account)	<input type="checkbox"/> Read	<input type="checkbox"/> Read <input type="checkbox"/> Write
Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers		

Access for other AWS accounts

No other AWS accounts associated with the resource.

[Add grantee](#)

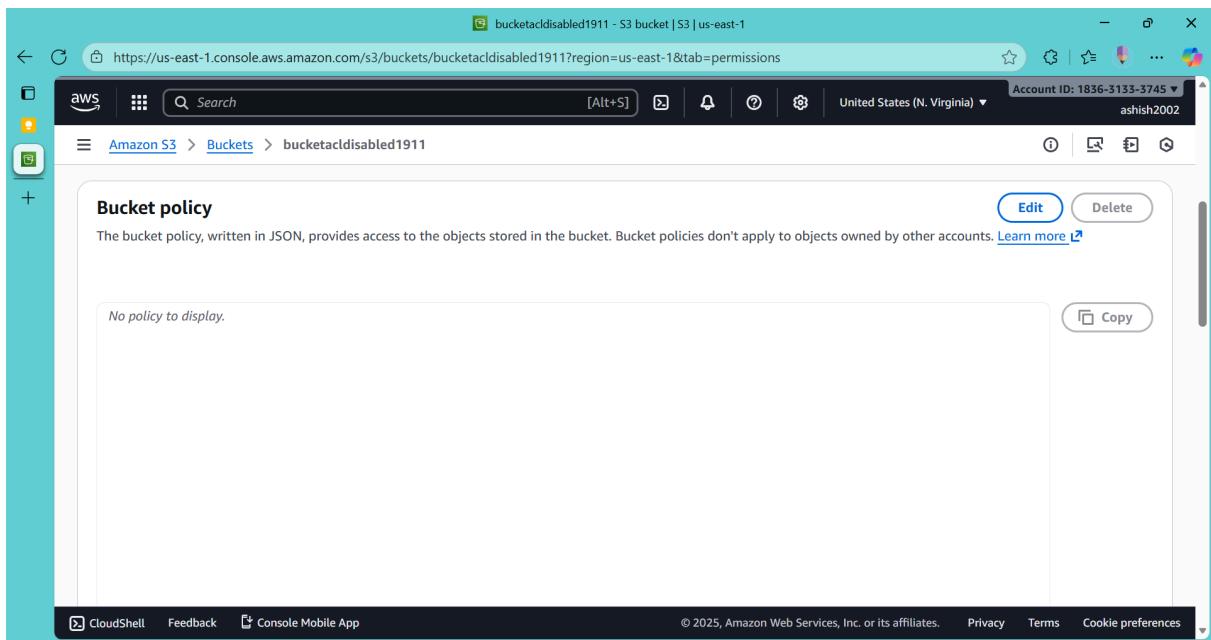
```
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestId>M0F1300P9PAZG0E</RequestId>
<HostId>JhILSBRx11b0RevmTVy2eS/exTpj9P/mnKYdeRwgMqFy296fdbY70gXbZhzDHJBMSxmVPwId7E</HostId>
</Error>
```

Creating ACL Disabled Bucket (Practical)

Follow the same steps above but choose **ACL disabled** when creating the bucket.

With ACL disabled you must use a **Bucket Policy** to control access.

Inside the bucket → Permissions → Bucket policy.



Example policy (fig 2.3):

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": "*",  
      "Action": ["s3:PutObject", "s3:GetObject"],  
      "Resource": "arn:aws:s3:::bucketacldisabled1911/*"  
    },  
    {  
      "Effect": "Deny",  
      "Principal": "*",  
      "Action": "s3:DeleteObject",  
      "Resource": "arn:aws:s3:::bucketacldisabled1911/*"  
    }  
  ]  
}
```

The screenshot shows the AWS S3 'Edit bucket policy' interface. The URL is https://us-east-1.console.aws.amazon.com/s3/bucket/bucketacldisabled1911/property/policy/edit?region=us-east-1. The policy JSON is:

```

1  {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Principal": "*",
7             "Action": ["s3:PutObject", "s3:GetObject"],
8             "Resource": "arn:aws:s3:::bucketacldisabled1911/*"
9         },
10        {
11            "Effect": "Deny",
12            "Principal": "*",
13            "Action": "s3:DeleteObject",
14            "Resource": "arn:aws:s3:::bucketacldisabled1911/*"
15        }
16    ]
17 }

```

The right panel shows a sidebar with 'Edit statement' and 'Select a statement' sections, and a button to 'Add new statement'.

ACL disabled gives minute-level control through policies. For example: allow uploading and reading, but deny delete.

Elements inside a policy

- **Effect** → Allow or Deny
- **Resource(s)** → which object(s) the policy applies to
- **Action(s)** → e.g., s3:GetObject, s3:DeleteObject, s3:PutObject
- **Principal** → who can access (IAM user, role, or for everyone)

Amazon Resource Name (ARN)

Used to identify AWS resources. Format:

```
arn:<partition>:<service>:<region>:<account-id>:<resourceType>/<resourceId>
```

S3 example (S3 is global):

```
arn:aws:s3:::bucketname
```

Partitions include aws, aws-cn, aws-us-gov.

Preventing object deletion — alternatives

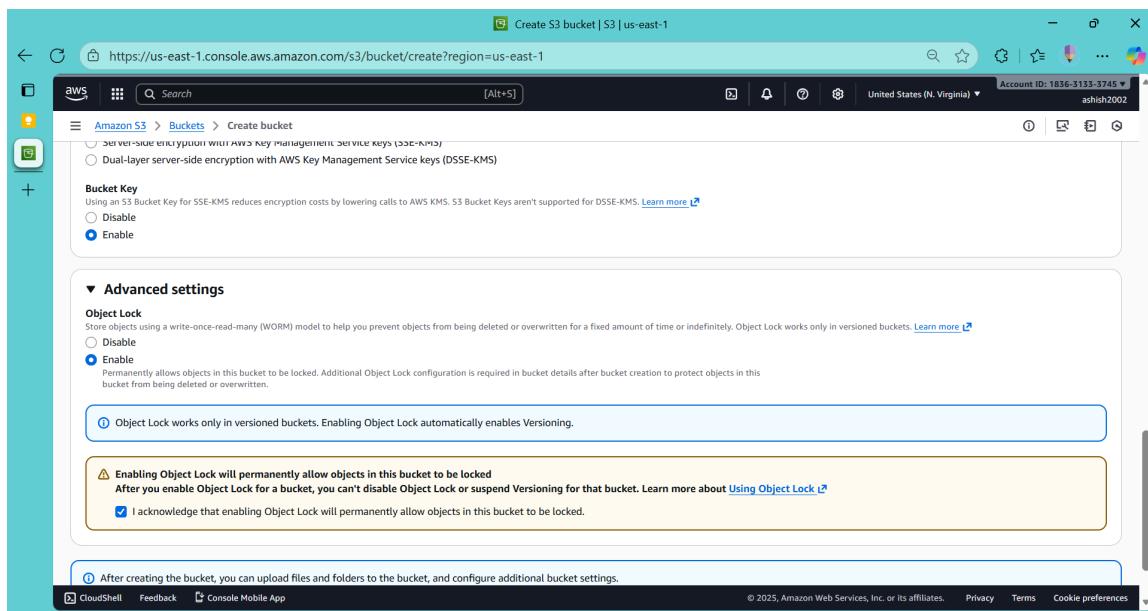
- Use **bucket policies** to `Deny s3:DeleteObject` for certain principals.
→ But if someone can change the policy, they could remove that deny, so lock down who can edit policies via IAM.
- Use **Object Lock** and **Versioning** for stronger protection (recommended for compliance or immutability).

Object Lock (enable at bucket creation)

Important: Object Lock must be enabled when you create the bucket. You cannot enable it later. Object Lock requires versioning.

Steps:

1. Enable **Object Lock** while creating the bucket.



2. Optionally set **Default Retention** rules.

Behavior notes:

- Enabling Object Lock automatically enables bucket **versioning**.
- Object Lock prevents objects from being deleted or overwritten according to retention rules.

Object Lock Modes & Cases

Case 1: Object Lock enabled, Default Retention disabled

- Behavior: Works like versioning. If someone deletes the object from the object list, a **delete marker** is created and older versions remain available. You can restore by removing the delete marker.

Case 2: Object Lock enabled, Default Retention enabled

- You must choose a retention mode:
 - **Governance mode** → Privileged IAM users (with special permissions) can bypass retention and delete during the retention period.
 - **Compliance mode** → No one can delete the object or remove retention until the retention period expires (strongest protection).

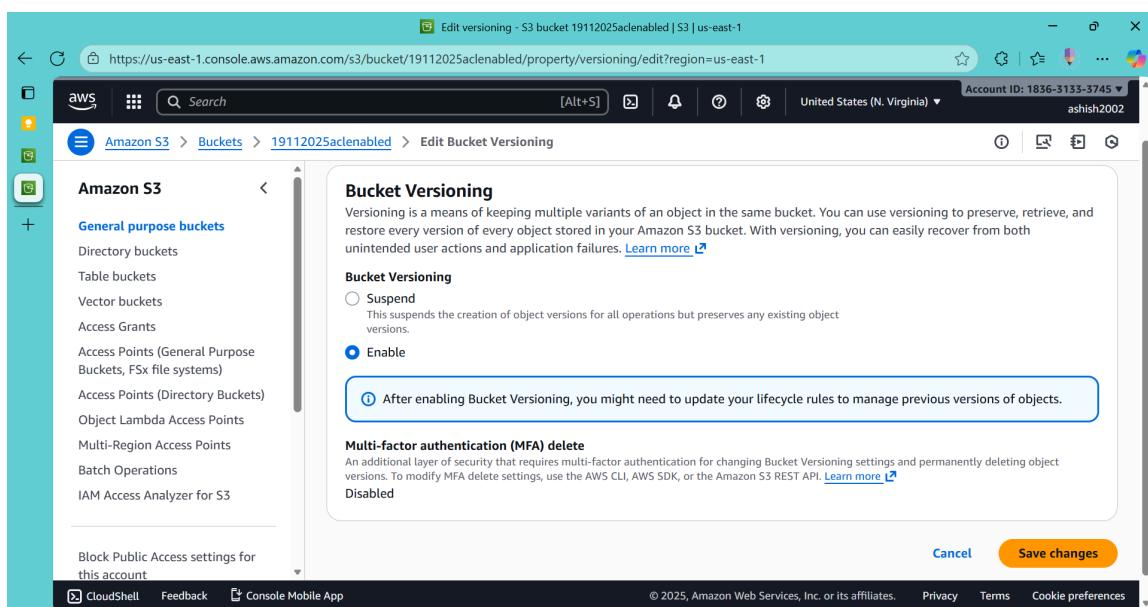
Use Compliance mode for strict immutability. Use Governance if you need controlled flexibility.

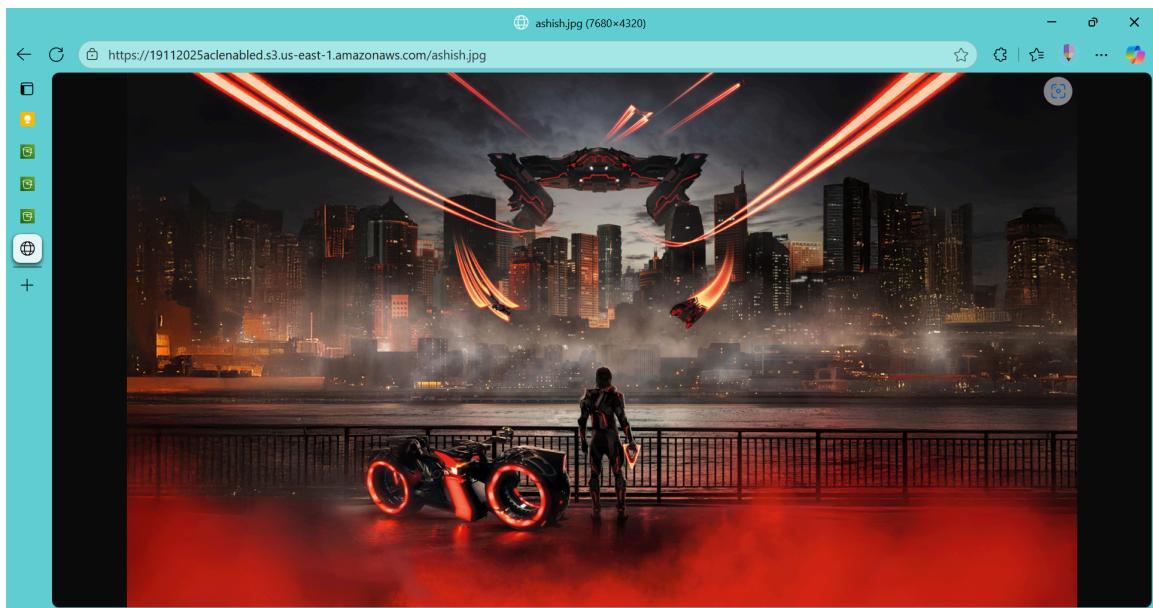
Bucket Versioning (how it works)

- **Versioning can be enabled anytime.**
- When versioning is on, the same object key can have multiple versions.

Example workflow:

1. Upload `ashish.png` → version V1.





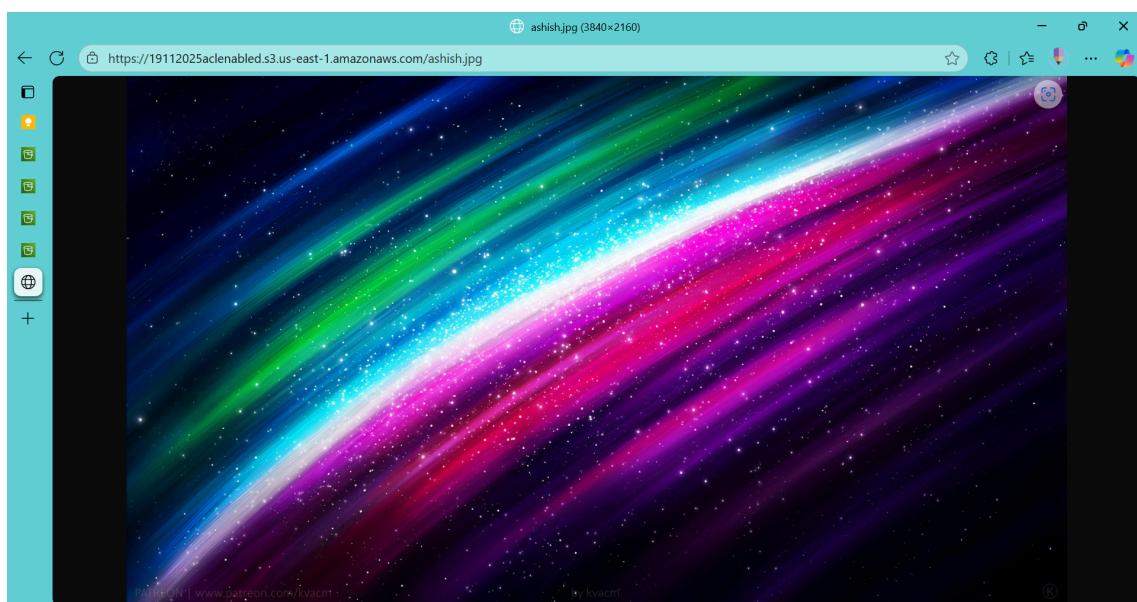
2. Upload another file with the same name [ashish.jpg](#) → new version V2; older V1 remains visible under versions.

A screenshot of the AWS S3 console showing the "Objects" tab for the "19112025aclenabled" bucket. The interface includes a search bar, filter options, and a table listing the uploaded files. The table has columns for Name, Type, Last modified, Size, and Storage class. Two objects are listed: "a.png" (Type: png, Last modified: November 19, 2025, 14:01:13 (UTC+05:30), Size: 696.3 KB, Storage class: Standard) and "ashish.jpg" (Type: jpg, Last modified: November 19, 2025, 17:03:07 (UTC+05:30), Size: 9.1 MB, Storage class: Standard).

Name	Type	Last modified	Size	Storage class
a.png	png	November 19, 2025, 14:01:13 (UTC+05:30)	696.3 KB	Standard
ashish.jpg	jpg	November 19, 2025, 17:03:07 (UTC+05:30)	9.1 MB	Standard

The screenshot shows the AWS S3 console interface. At the top, a header bar displays the URL <https://us-east-1.console.aws.amazon.com/s3/upload/19112025aclenabled?region=us-east-1>, the account ID 1836-3133-3745, and the region United States (N. Virginia). The main area is titled "Upload" and shows a table of files and folders. One file, "ashish.jpg", is listed with a size of 3.2 MB and type image/jpeg. Below the table is a section titled "Destination" with a dropdown menu set to "Destination". At the bottom of the page, there are links for CloudShell, Feedback, and Console Mobile App, along with copyright information for 2025, Amazon Web Services, Inc. or its affiliates.

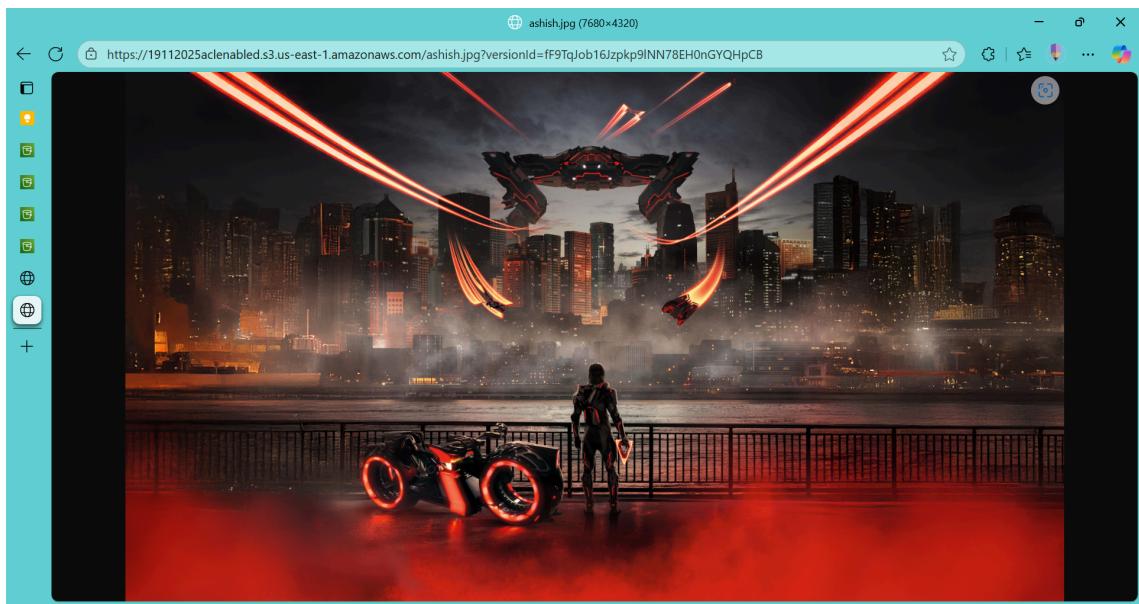
3. The object URL without a version shows the **latest version**.



4. To access an older version:

bucketname.region.amazonaws.com/ashish.jpg?versionId=4344

(Each version has a **versionId**.) (fig 3.1)



Delete behavior with versioning:

- Deleting the object from the object list adds a **delete marker**; the object disappears from the normal list but older versions are still stored.

Summary	Successfully deleted	Failed to delete
Source s3://19112025aclenabled	1 object, 3.2 MB	0 objects

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with various bucket categories like 'General purpose buckets', 'Table buckets', etc. The main area is titled '19112025aclenabled' and shows a single object named 'a.png'. The object details are: Name: a.png, Type: png, Last modified: November 19, 2025, 14:01:13 (UTC+05:30), Size: 696.3 KB, Storage class: Standard.

- To restore the object in the object list, remove the delete marker (it acts like a recycle bin).

The screenshot shows the AWS S3 console interface with the 'Objects' list. One object, 'ashish.png', is highlighted and identified as a 'Delete marker'. The other object listed is 'a.png'. The table headers are Name, Type, Version ID, Last modified, Size, and Storage class.

Name	Type	Version ID	Last modified	Size	Storage class
<input checked="" type="checkbox"/> ashish.png	Delete marker	S9TH.Tflt4A3Q2 nXkiGVsoPqeYB arjH6	November 19, 2025, 16:57:43 (UTC+05:30)	0 B	-
<input type="checkbox"/> a.png	png	71R2vMU2imz WFDPSB5JQFkX p_wShPRvd	November 19, 2025, 16:57:21 (UTC+05:30)	203.7 KB	Standard

- Deleting a specific version from the versions view permanently removes that version.
- If **versioning is disabled**, uploading a file with the same name replaces the old file (no version history).

API note: To read object versions via API you may need [s3:GetObjectVersion](#) permission.

S3 Static Website Hosting

- Website URL format:

<bucket_name>.s3-website.<region>.amazonaws.com

- S3 static websites are **HTTP only**. To serve over **HTTPS**, put **CloudFront** in front of S3.

Practical

1. Upload `index.html` to your bucket.

The screenshot shows the AWS S3 console interface. At the top, a green success message box displays "Upload succeeded" and "For more information, see the Files and folders table." Below this, a summary table shows the upload results:

Destination	Succeeded	Failed
s3://19112025aclenabled	1 file, 42.0 B (100.00%)	0 files, 0 B (0%)

Below the summary, a table lists the uploaded file:

Name	Folder	Type	Size	Status	Error
index.html	-	text/html	42.0 B	Succeeded	-

At the bottom of the page, there are links for CloudShell, Feedback, and Console Mobile App, along with copyright and legal information.

2. Enable static website hosting from bucket properties.

The screenshot shows the "Static website hosting" section of the AWS S3 bucket properties. It includes a recommendation for AWS Amplify Hosting and a status message for S3 static website hosting.

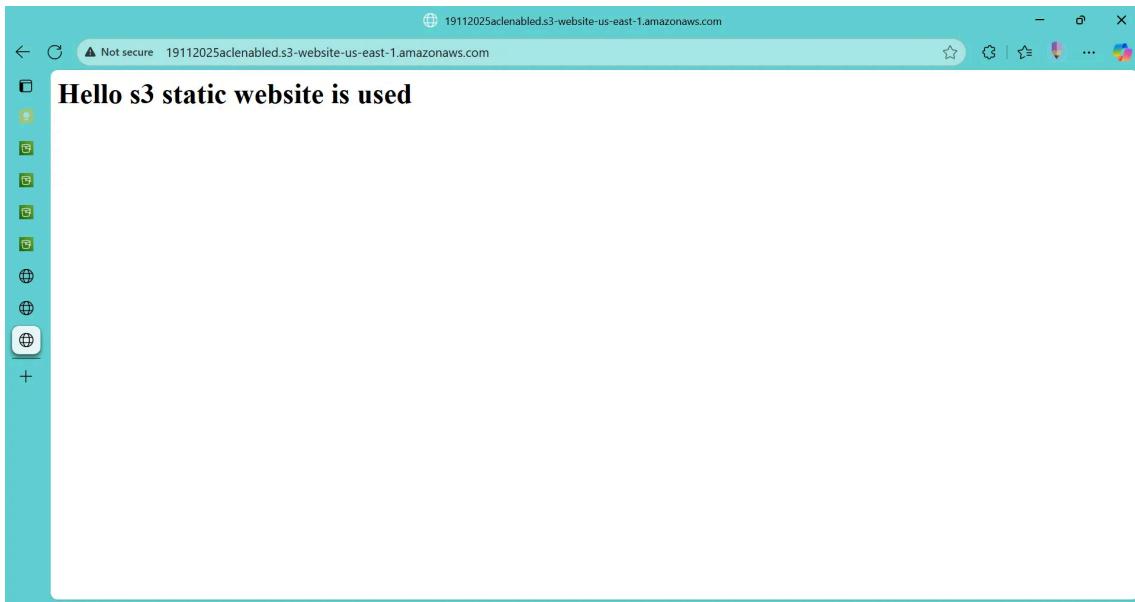
Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

We recommend using AWS Amplify Hosting for static website hosting
Deploy a fast, secure, and reliable website quickly with AWS Amplify Hosting. Learn more about [Amplify Hosting](#) or [View your existing Amplify apps](#)

S3 static website hosting
Disabled

3. You will get a website URL to open.



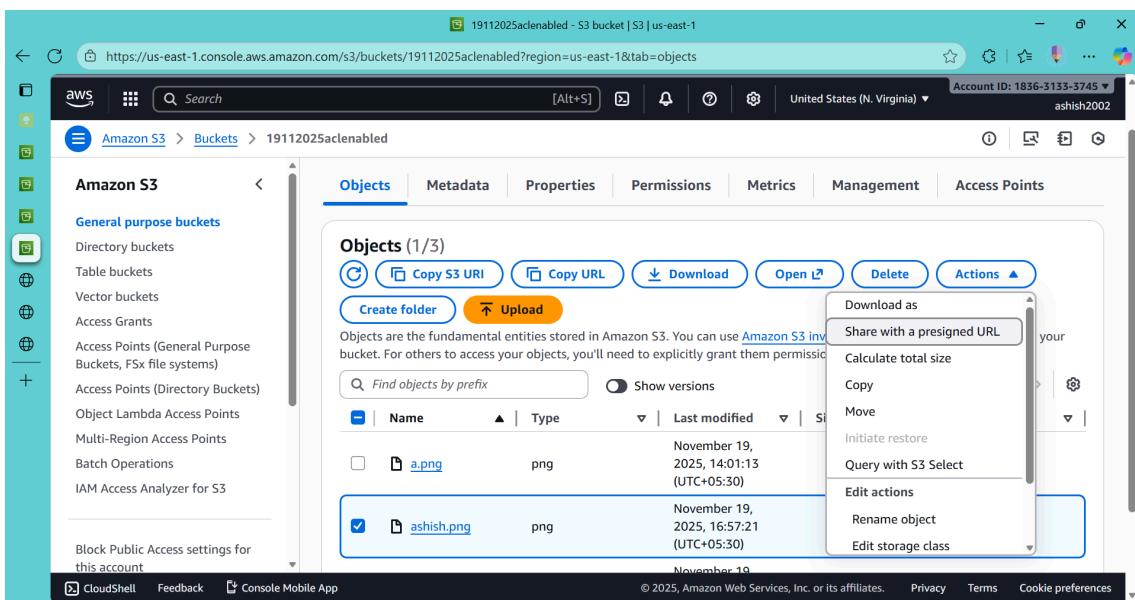
It's recommended to use S3 (with CloudFront for HTTPS) for static websites instead of running EC2 for static hosting.

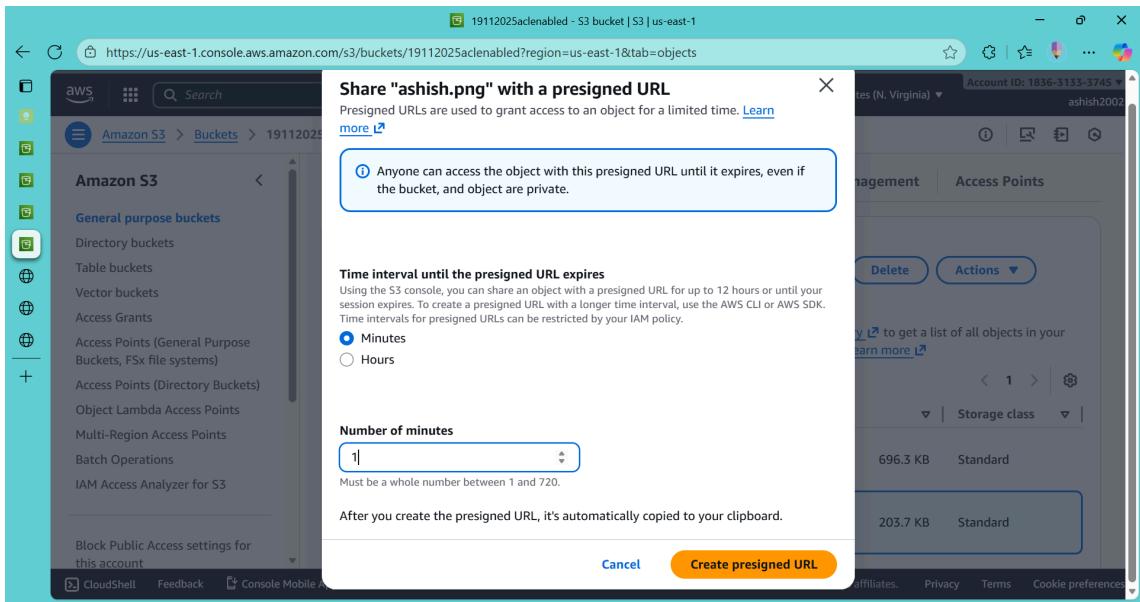
Pre-signed URLs

- A **pre-signed URL** is a temporary link that expires after a set time.
- Common use: give time-limited access for streaming or downloads.

Practical

- In the Console: select `ashish.jpg` → Actions → Share with a presigned URL.





- Example: generate a URL that is valid for **1 minute** — after that it will expire.

MFA Delete (Multi-Factor Authentication for versioning)

- **Requirement:** Bucket versioning must be ON.
- Steps:
 1. Enable MFA device in AWS Security Credentials (authenticator app or hardware).
 2. Note the MFA device ARN.
 3. Use AWS CLI with configured credentials to enable MFA Delete.

Example CLI command:

```
aws s3api put-bucket-versioning --bucket bucket1711202555 \
--versioning-configuration Status=Enabled,MFADelete=Enabled \
--mfa "arn:aws:iam::183631333745:mfa/Authapp 123456"
```

Ways to implement MFA:

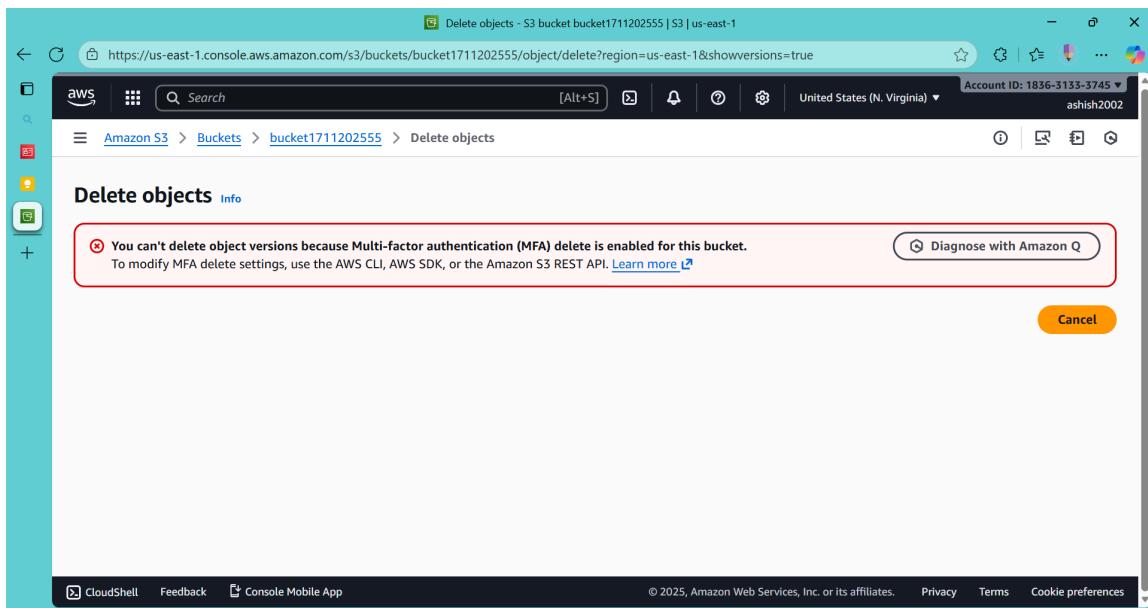
- Software TOTP (authenticator app)
- YubiKey (hardware)
- Other hardware TOTP devices

After enabling MFA Delete:

- Deleting from the main object list still puts a delete marker (versioning behavior).
- Deleting a **specific version** or removing delete markers requires MFA — extra protection.

Name	Type	Version ID	Last modified	Size	Storage class
a.png	png	oXQKFh.HPdcy57A	12:22:40 (UTC+05:30)	696.3 KB	Standard
Screenshot 2025-11-04 184741.png	Delete marker	uduBdHvU6chufrM1AJ	November 19, 2025, 13:31:34 (UTC+05:30)	0 B	-
Screenshot 2025-11-04 184741.png	png	1UD1rnEX0AlgGwX8igtHZmecRsg6zyHz	November 19, 2025, 13:31:05 (UTC+05:30)	203.7 KB	Standard
Screenshot 2025-11-04 184741.png	Delete marker	8NvzwSt.EhyZcaoa16EyXPLdQACsrXH	November 19, 2025, 12:56:33 (UTC+05:30)	0 B	-
Screenshot 2025-11-04 184741.png	png	null	November 17, 2025, 15:31:39 (UTC+05:30)	203.7 KB	Standard

Name	Type	Version ID	Last modified	Size	Storage class
a.png	png	.XQ7XUyes4JtWAMRM	November 18, 2025, 12:22:40 (UTC+05:30)	696.3 KB	Standard
Screenshot 2025-11-04 184741.png	Delete marker	uduBdHvU6chufrM1AJ	November 19, 2025, 13:31:34 (UTC+05:30)	0 B	-
Screenshot 2025-11-04 184741.png	png	1UD1rnEX0AlgGwX8igtHZmecRsg6zyHz	November 19, 2025, 13:31:05 (UTC+05:30)	203.7 KB	Standard
Screenshot 2025-11-04 184741.png	Delete marker	8NvzwSt.EhyZcaoa16EyXPLdQACsrXH	November 19, 2025, 12:56:33 (UTC+05:30)	0 B	-
Screenshot 2025-11-04 184741.png	png	null	November 17, 2025,		



Final practical reminders

- Enable **Versioning** to keep older copies and recover from accidental deletes.
- Enable **Object Lock** at bucket creation for immutable storage (compliance needs).
- Use **Bucket Policies** and **IAM** to restrict who can edit policies and who can delete objects.
- Use **Pre-signed URLs** for temporary, time-limited access.
- Use **CloudFront** if you need HTTPS for a static S3 website.

Thank you.