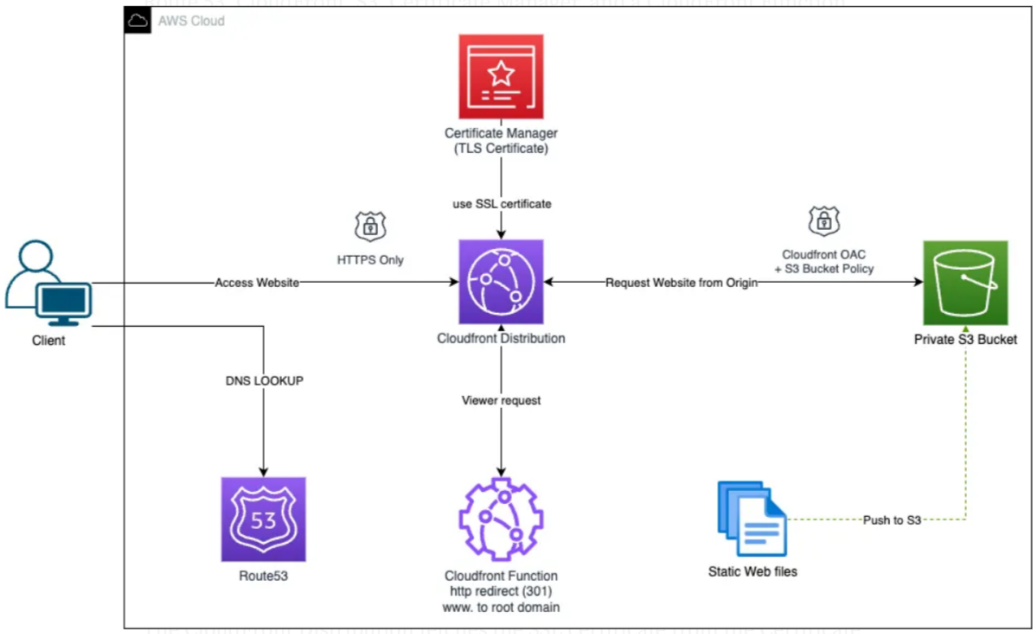
Project 2

Deploy a Static Website on AWS

What is a static website?

A static site is built with pages of static content or plain HTML, Javascript, or CSS code. This content stays consistent regardless of the user or where the user is and offers little interactivity. Because the content served is not dependent on a particular user, it can be pre-rendered, leading to fast performance and lightweight overhead in simple use cases.

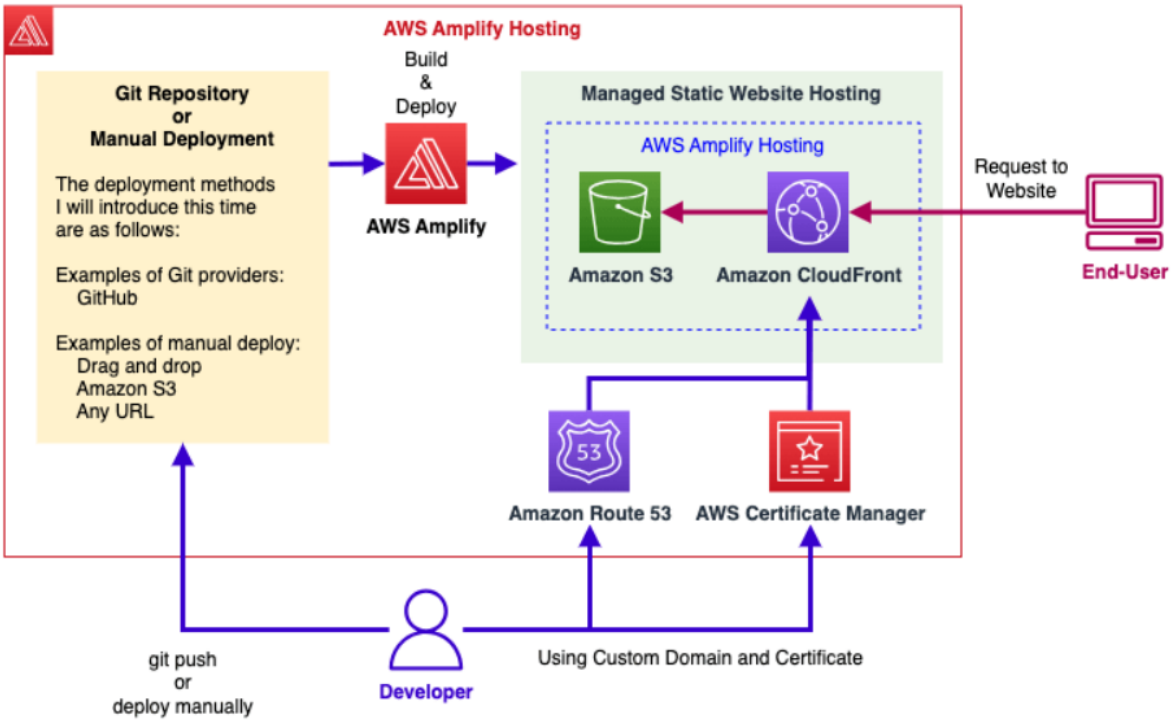


1 TableofContents

Introduction

As enterprises become more digital operations, their websites span a wide spectrum, from mission-critical e-commerce sites to departmental apps, and from business-to-business (B2B) portals to marketing sites. Factors such as business value, mission criticality, service level agreements (SLAs), quality of service, and information security drive the choice of architecture and technology stack.

The simplest form of website architecture is the *static website*, where users are served static content (HTML, images, video, JavaScript, style sheets, and so on). Some examples include brand microsites, marketing websites, and intranet information pages. Static websites are straightforward in one sense, but they can still have demanding requirements in terms of scalability, availability, and service-level guarantees. For example, a marketing site for a consumer brand may need to be prepared for an unpredictable onslaught of visitors when a new product is launched.



Architecture and user access flow with AWS Amplify Hosting in the AWS Amplify Console

Overview

In this tutorial, you will learn how to deploy a static website with AWS Amplify. Amplify offers a Git-based CI/CD workflow for building, deploying, and hosting websites. Static websites deliver HTML, JavaScript, images, video and other files to your website visitors. Static websites are very low cost, provide high-levels of reliability, require almost no IT administration, and scale to handle enterprise-level traffic with no additional work.

Key differences between static vs. dynamic websites

Static and dynamic websites serve different purposes, and each has distinct advantages and disadvantages.

Think of static websites as snapshots of content that remain unchanged unless manually updated. They work well in some instances but can limit opportunities for interactivity and personalization.

Dynamic websites can change based on user input or real-time data, creating a more interactive and personalized experience. However, they can be more complex to develop and maintain and may experience performance issues if not optimized properly.

FEATURE	STATIC WEBSITES	DYNAMIC WEBSITES
Content	Remains fixed	Can change based on user input or real-time data
Development	Typically simpler	More complex, requiring server-side scripting and databases
Performance	Generally faster	Can be slower, especially with complex interactions or large databases
Security	Generally more secure	Can be less secure due to potential vulnerabilities in server-side code and databases
Cost	Often more cost-effective	Can be more expensive to develop and maintain
Use cases	Simple websites, brochures, landing pages	E-commerce stores, social media platforms, news websites

Objective

1. Blogs and Personal Websites: For bloggers and individuals looking to establish an online presence, static websites offer an easy-to-manage platform. The focus is often on content rather than complex features, making static sites a suitable choice.
2. Landing Pages and Portfolios: Businesses and professionals showcasing their work or products can benefit from the fast-loading pages of static sites. This efficiency enhances user experience, keeping potential clients engaged.
3. Documentation and Informational Sites: When the primary goal is to provide information that does not require frequent updates, a static website is an excellent choice. Its reliability and speed ensure that users can access the needed information without delays.
4. Event or Promotional Pages: For events or promotions, where the content is temporary and does not require dynamic updates, static pages are ideal. They provide all the necessary information in a format that is easy to access and share, even on social media.

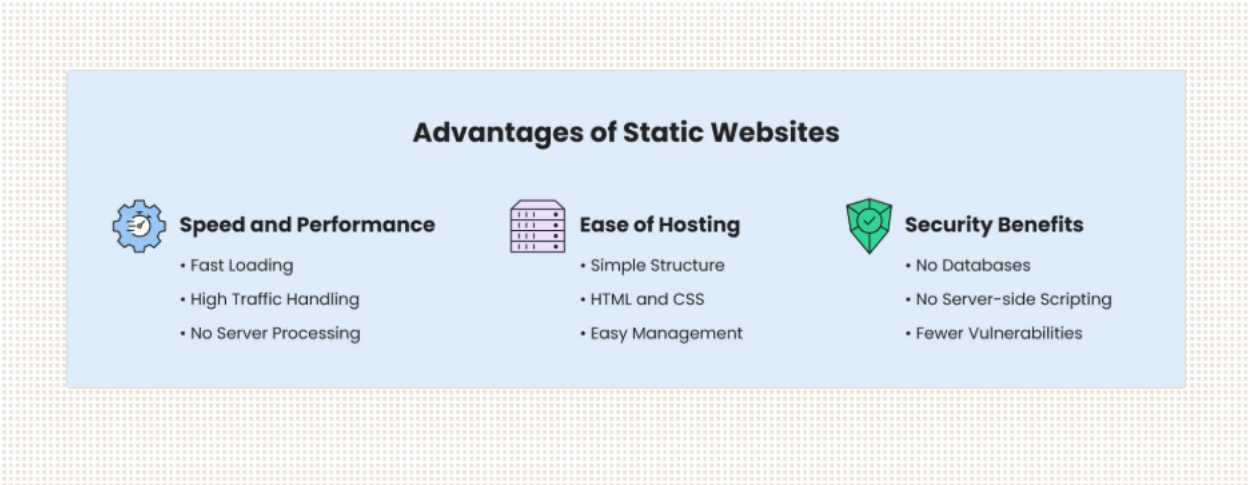
Prerequisites

Before starting this tutorial, you will need:

- An **AWS account**: if you don't already have one follow the Setup Your Environment tutorial.
- Your **AWS profile** configured for local development.
- **Installed** on your environment: Nodejs and npm.
- Familiarity with git and a Github account.

Advantages of Static Websites

There are several upsides of static web pages, and we'll outline them in this section. They are mostly related to their speed and performance. Plus, static sites are more secure and easier to set up in your web hosting.



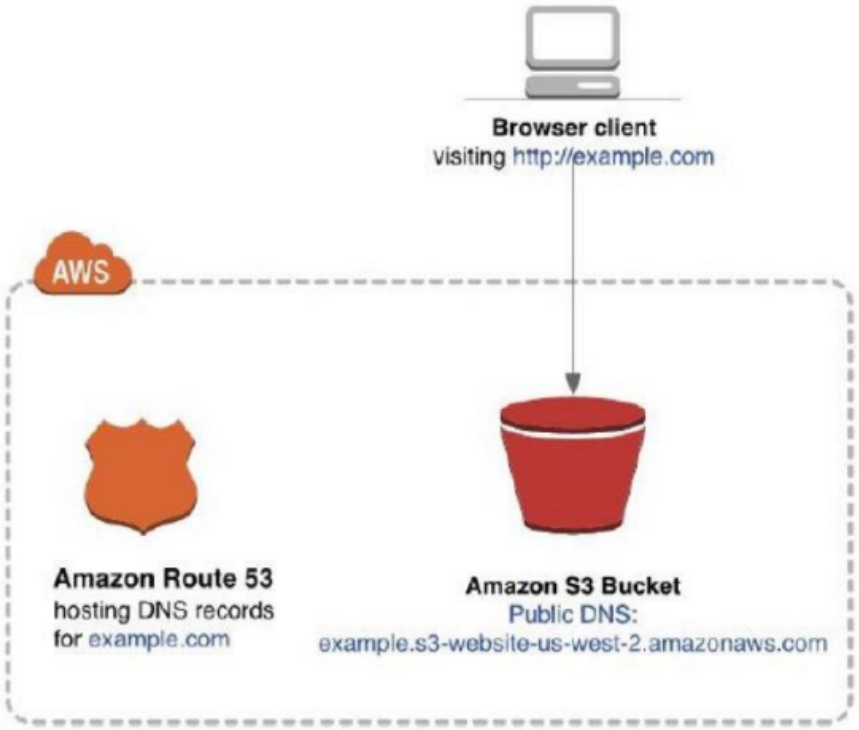
Implementation

Hosting a static website using Amazon S3

Amazon Simple Storage Service (Amazon S3) can host static websites without a need for a web server. The website is highly performant and scalable at a fraction of the cost of a traditional web server. Amazon S3 is storage for the cloud, providing you with secure, durable, highly scalable object storage. A simple web services interface allows you to store and retrieve any amount of data from anywhere on the web.[1]

You start by creating an Amazon S3 bucket, enabling the Amazon S3 website hosting feature, and configuring access permissions for the bucket. After you upload files, Amazon S3 takes care of serving your content to your visitors.

Amazon S3 provides HTTP web-serving capabilities, and the content can be viewed by any browser. You must also configure Amazon Route 53, a managed Domain Name System (DNS) service, to point your domain to your Amazon S3 bucket. Figure 3 illustrates this architecture, where *http://example.com* is the domain.

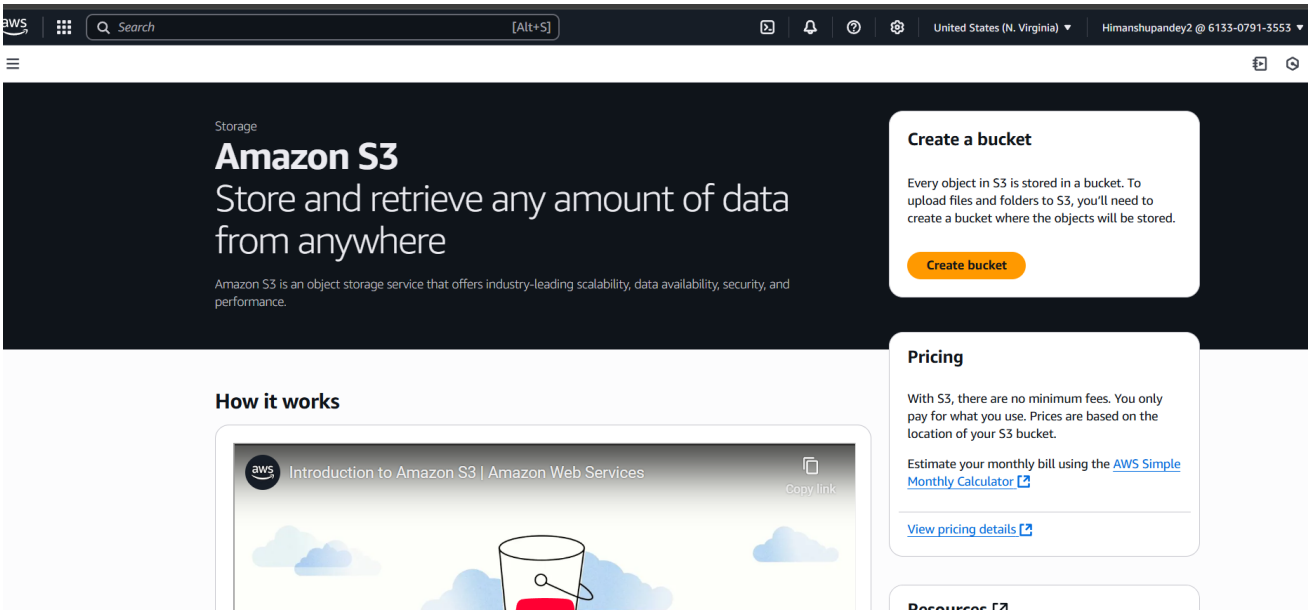


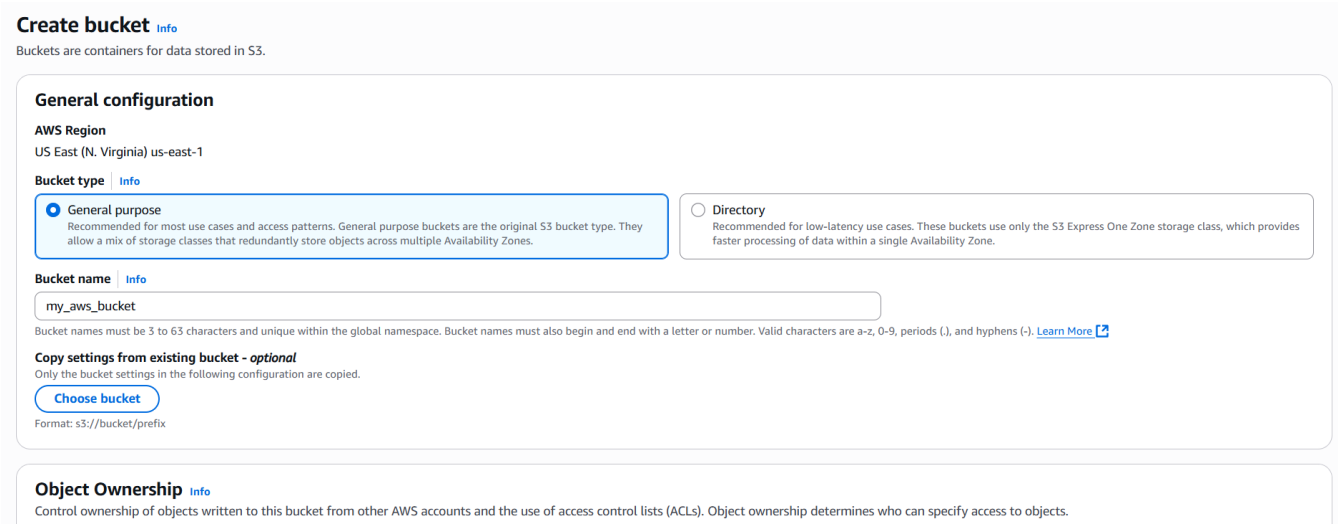
Task 1: Sign in to AWS Management Console

1. Click on the **Open Console** button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,

- Leave the Account ID as default. Never edit/remove the 12-digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
- Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign in** button.

3. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1.**





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.

Object Ownership
Bucket owner enforced

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 all. 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**

⚠ **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.

- [x] 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

Bucket Versioning

- (●) Disable
- ( ) Enable

Tags - *optional* (0)

You can use bucket tags to track storage costs and organize buckets. Learn more

No tags associated with this bucket.

Add tag

Default encryption Info

Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type Info

- (●) Server-side encryption with Amazon S3 managed keys (SSE-S3)
- ( ) Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- ( ) Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)
  Secure your objects with two separate layers of encryption. For details on pricing, see **DSSE-KMS pricing** on the **Storage** tab of the Amazon S3 pricing page.

Bucket Key

Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. Learn more

- ( ) Disable
- (●) Enable

▶ **Advanced settings**

ⓘ After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

Cancel | Create bucket

Task 2: Creating a S3 Bucket

In this task, we are going to create a new S3 bucket in the US East (N. Virginia) region with a unique name disabling ACLs, and allowing public access for hosting the static website.

1. Navigate to **S3** by clicking on the **Services** menu at the top, then click on **S3** in the **Storage** section.
2. In the S3 dashboard, click on the **Create Bucket** button.
3. In the General Configuration, **Bucket name**: Enter **abcxyz**

- **Note:** S3 Bucket names are globally unique, choose a name that is available. Maybe you can enter your name and create one.

4. AWS Region: Select **US East (N. Virginia) us-east-1**
5. Object ownership: Select **ACLs disabled (recommended)** option
6. In the option of **Block Public Access settings for this bucket**, **Uncheck** the option of **Block all public access**.

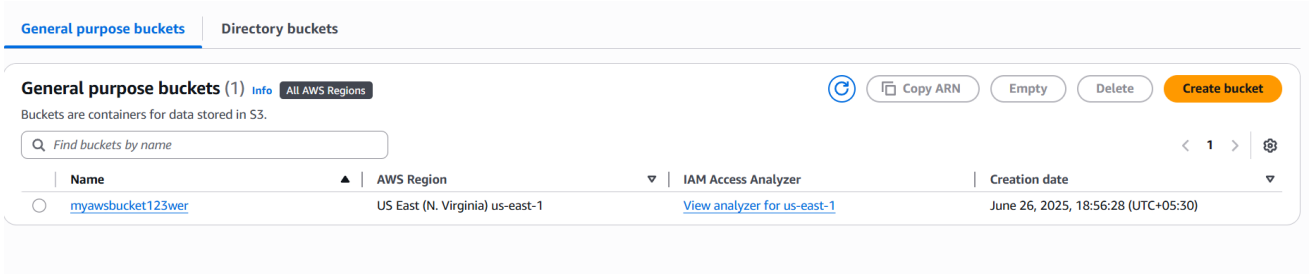
- **Check** the I acknowledge that the current settings might result in this bucket and the objects within becoming public checkbox.

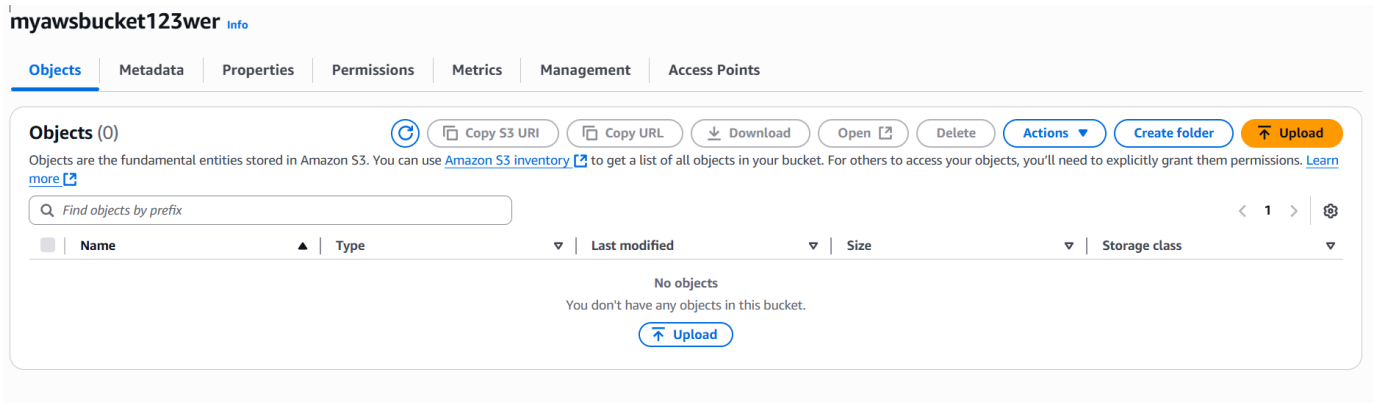
7. Keep everything default and click on **Create Bucket** button.

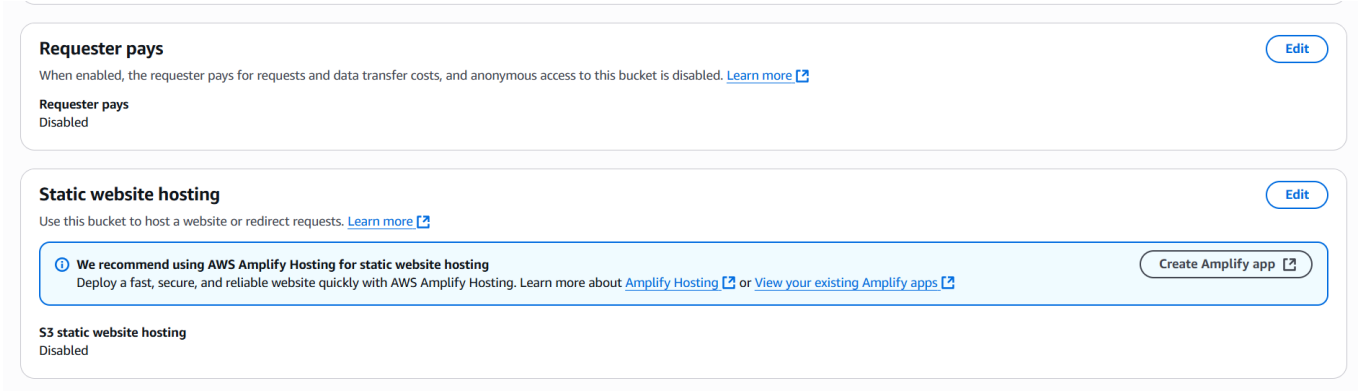
Task 3: Enable Static Website Hosting settings

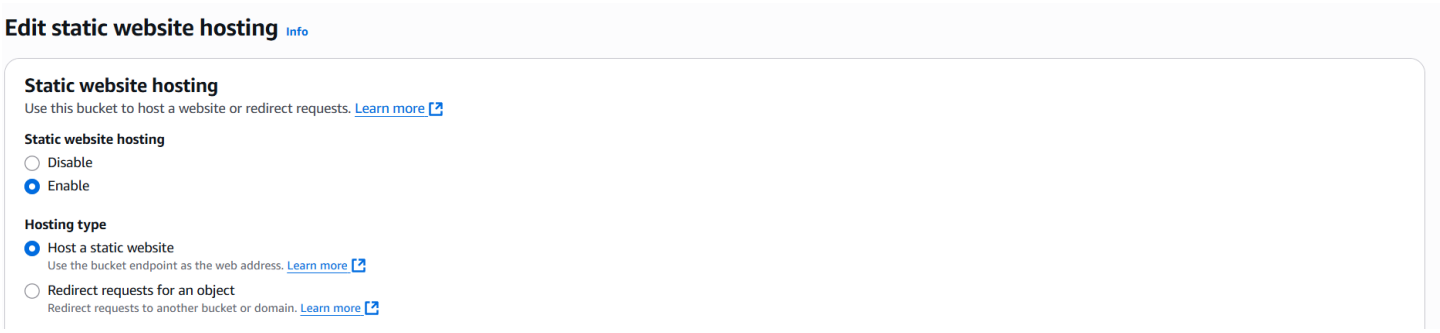
In this task, we will enable static website hosting for our S3 bucket, configure it to use index.html and error.html, copy the provided endpoint, upload two files, and configure the bucket policy by copying its ARN and pasting the provided policy code.

1. To proceed, go to the **S3 bucket name** that you created and click on it. After that, navigate to the **Properties** tab which can be found at the top of the screen.
2. Scroll down to the **Static website hosting** section and click on **Edit** button.




myawsbucket123wer Info
Objects
Metadata
Properties
Permissions
Metrics
Management
Access Points
Objects (0)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
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
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
No objects
You don't have any objects in this bucket.
Upload


Requester pays
Edit
When enabled, the requester pays for requests and data transfer costs, and anonymous access to this bucket is disabled. Learn more
Requester pays
Disabled
Static website hosting
Edit
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
Create Amplify app
S3 static website hosting
Disabled

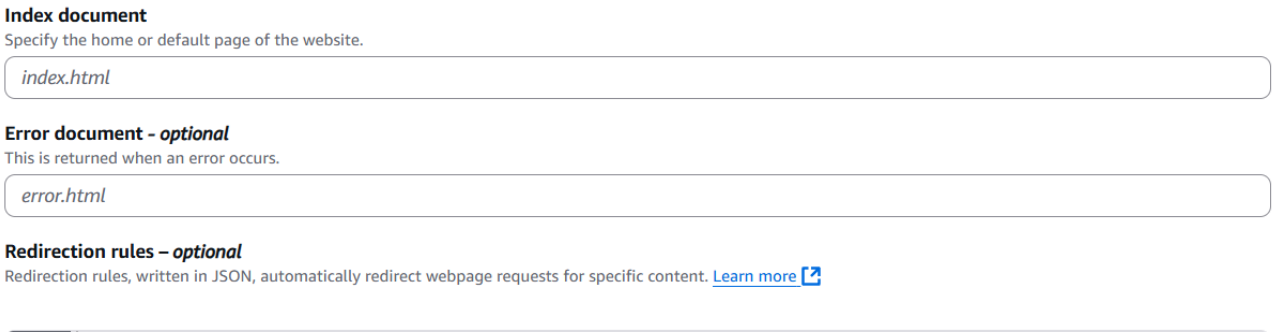

Edit static website hosting Info
Static website hosting
Use this bucket to host a website or redirect requests. Learn more
Static website hosting
Disable
Enable
Hosting type
Host a static website
Use the bucket endpoint as the web address. Learn more
Redirect requests for an object
Redirect requests to another bucket or domain. Learn more

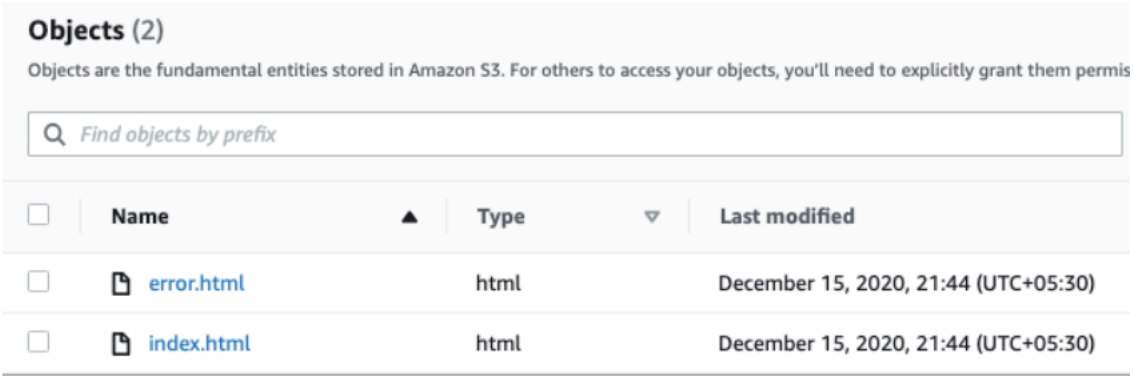
3. In the **Static website hosting** dialog box

- Static website hosting: Select **Enable**
- Hosting type: Choose **Host a static website**
- Index document: Type ***index.html***
- Error document: Type ***error.html***
- Click on **Save Changes**.

4. Go to the **Properties** tab of your S3 bucket, and find the **Static website hosting** section. **Copy** the Endpoint provided in this section to your clipboard and **save** it for future reference.

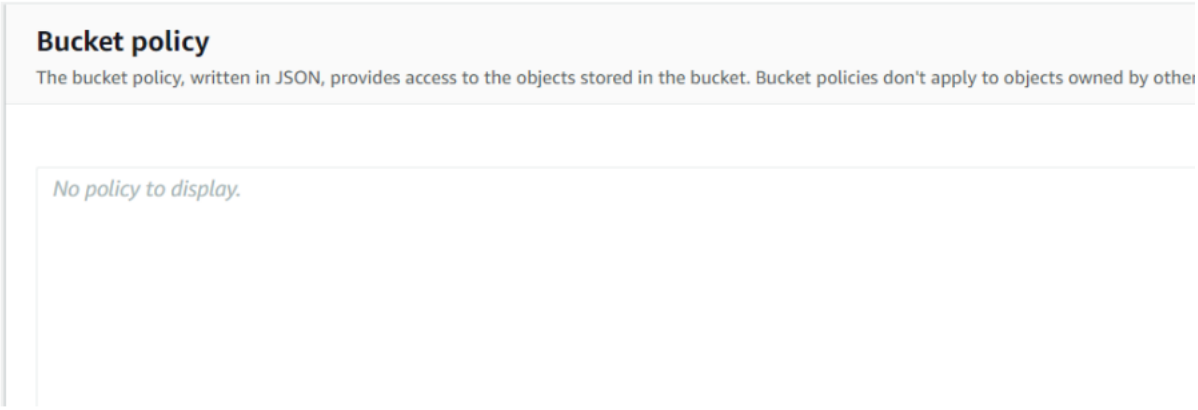
5. The next step is to download the zip file by clicking on the link, extract it, and upload two files named **index.html** and **error.html** to the S3 bucket you created earlier.





6. To configure your S3 bucket, access the **Permissions** tab and make the necessary configurations.

- In the **Permissions** tab, Click on the **Edit** button beside the **Bucket Policy**.



- You will be able to see a Blank policy editor.
- Before creating the policy, you will need to copy the **ARN** (Amazon Resource Name) of your bucket.
- Copy the **ARN** of your bucket to the clipboard. It is displayed at the top of the policy editor. it will look like **ARN:"arn:aws:s3:::your-bucket-name**".
- In the policy below, **Update** the bucket ARN on the Resource key value and **paste** the below policy code in the editor.

```
{"Version":"2012-10-17",
 "Statement":[{
   "Sid":"PublicReadGetObject",
   "Effect":"Allow",
   "Principal": "*",
   "Action":["s3:GetObject"],
   "Resource":["arn:aws:s3:::S3_BUCKET_NAME_GOES_HERE/*"]
 }]
}
```

Click on **Save changes** button.

Task 4: Test the website

1. Now copy the **static website URL** (that we saved earlier) and run it in your browser. You will be able to see the index.html file's text. A sample screenshot is attached below: