Personal Blog on IBM Cloud Static Web Apps

BY:

Hariprasath.D

PHASE 2: Innovation

Project Definition:

The project is to create a personal travel blog hosted on IBM Cloud Static Web Apps. The goalis to share travel adventures, tips, and captivating photos to inspire others to explore the worldand create unforgettable memories. This involves designing the blog structure, creating engaging content, setting up the IBM Cloud Static Web Apps, and ensuring ease of updating the blog.

Abstract:

Our Project is a captivating personal travel blog hosted on IBM Cloud Static Web Apps. This blog is an immersive window into the adventures, insights, and captivating moments experienced by an avid explorer. From the pristine beaches of Bali to the bustling streets of Tokyo, every blog post is a narrative that takes readers on a visual journey through picturesque destinations and shares invaluable travel tips.

The blog's IBM Cloud Static Web Apps hosting ensures seamless updates, making it easyfor the author to chronicle their ongoing adventures. With engaging content, stunning photography, and insightful travel guides, Our Project inspires readers to embark on their own journeys and discover the beauty of the world.

Explore the world through the eyes of a passionate traveler, and let our project be yourguide to forging unforgettable memories and igniting your own wanderlust.

There are 2 major steps in our Project:

- Creation and Designing of the Blog
- Deployment of the blog using IBM cloud services

Creation and Designing of the Blog:

→ As Initial step, We will be creating a blog which focuses on sharing the experience and igniting the wanderlust spirit within people by sharing ourexperience as a form of blog which can reach wider range of audiences.

→ In the next step we will be creating our blog using a blogging platform eitherwith wordpress.com or blogger.com.

→ As the next step we will be registering our domain name as needed for betteraccess throughout the internet.

→ After setting up my blog and creating the blog pages as required,we will beuploading or induce our content within the blog.

→ Then we have created social media profiles for our blog on platforms like Facebook, Twitter, Instagram, and Pinterest. We will share our blog posts andengage with our audience.

→ Now our blog is designed and ready for deployment.

Deployment of the blog using IBM cloud services:

Since our blog is a Static website we will carry out the process of hosting ourBlog according to that method.

Hosting static websites on IBM Cloud uses Cloud Object Storage (COS) and Cloud Internet Services (CIS) (with Page Rules and Edge Functions). These services provide the following features needed to serve static websites.

- **Auto-serving static assets from provider-managed HTTP service (CloudObject Storage).**
- **Custom domain support to serve content from user-controlled domainname (CIS - Page Rules).**
- **Configurable Index and Error documents (CIS - Edge Functions).**

Here are the steps needed to host a static website on IBM Cloud by combiningthose services.

Serving static assets

IBM Cloud Object Storage is a scalable storage solution for cloud applications.Files are managed through a RESTful HTTP API and stored in user-defined collections called “buckets”. Bucket files are returned as HTTP responses fromHTTP GET requests.

COS supports an optional “*anonymous read-only access*” setting for buckets. This means all files in the bucket will be accessible using anonymous HTTP GET requests.

Putting HTML, CSS and JS files in a public bucket allows static websites to be served directly by COS. Users are charged for bandwidth used and HTTP requests

received for all bucket files.

Create IBM Cloud Object Storage instance

If you already have an instance of Cloud Object Storage you can skip this step…

- Provision a new instance of IBM Cloud Object Storage

Create IBM Cloud Object Storage Bucket

- Open the COS instance from the Resource List.
- Create a new COS bucket to host the static site files.
    - Choose a Bucket name
    - Choose the `Resiliency, Location` and `Storage Class` options for the bucket.

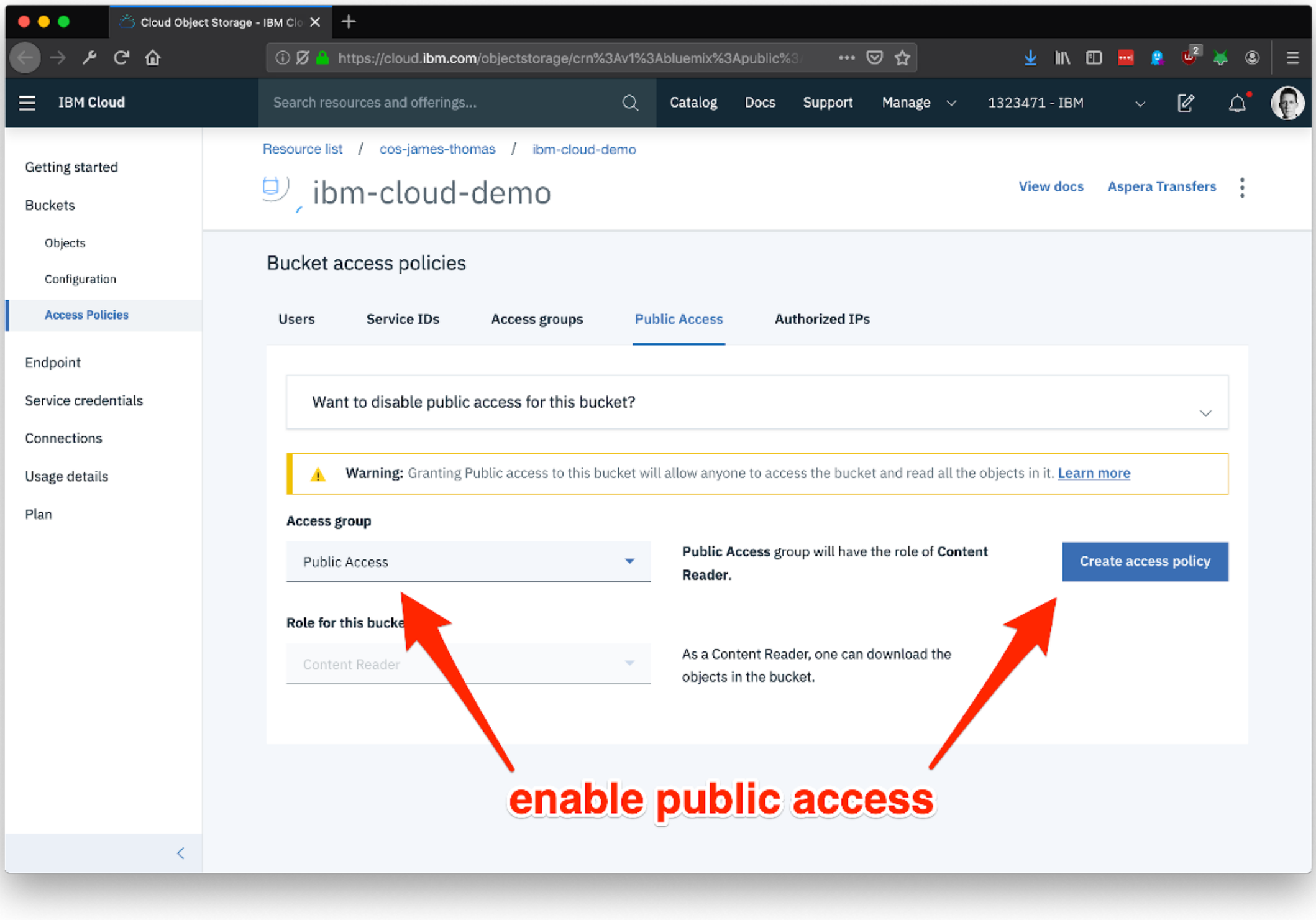
Any choices for these options can be used - it does not affect the static site hosting capability. For more details on what they mean, please see this documentation.

Upload Static Assets To Bucket

- Upload static file assets to the new bucket.

Enable Public Access to bucket files

- Click the *"Access Policies"* menu item from the bucket level menu.
- Click the "*Public Access*" tab from the bucket access policy page.
- Check the Access Group drop-down has "*Public Access*" option selected.
- Click the "*Create access policy*" and then "*Enable*" on the pop menu.

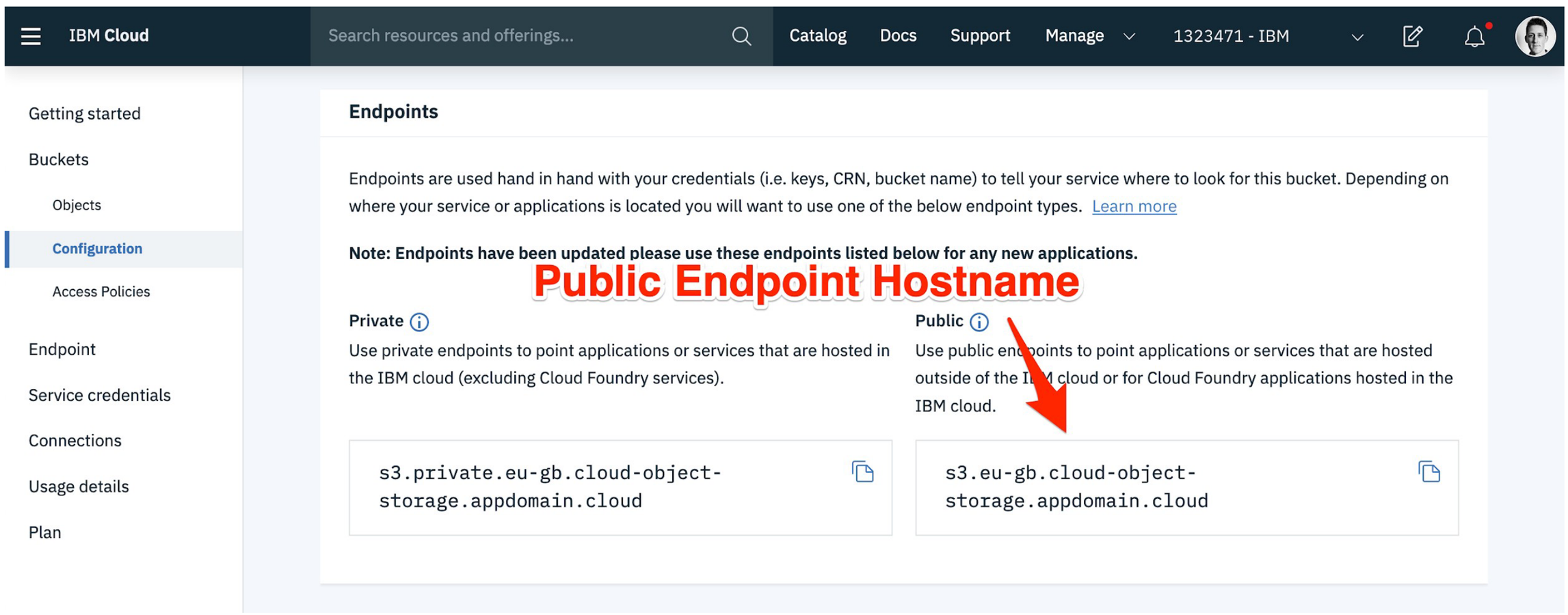


Check bucket files are accessible

Bucket files should now be accessible using the service endpoint URL, bucket id and file names. COS supports providing the bucket name in the URL path or a sub-domain on the service endpoint.

- Open the "*Configuration*" panel on the bucket page.
- Retrieve the **public endpoint** shown, e.g.

  `s3.<REGION>.cloud-object-storage.appdomain.cloud`



Bucket files (like `index.html`) should now be accessible by a web browser. COS supports both HTTP and HTTPS traffic. Bucket files are available using the following URLs.

vhost addressing

```
<BUCKET_NANME>.s3.eu-gb.cloud-object-storage.appdomain.cloud/index.html
```

url path addressing

```
s3.<REGION>.cloud-object-storage.appdomain.cloud/<BUCKET_NANME>/index.html
```

Bucket files can now be referenced directly in external web applications. COS buckets are often used to store large application assets like videos or images. **For hosting an entire website, it is often necessary to serve content from a custom domain name, rather than the COS bucket hostname.**

Custom domain support

Cloud Internet Services Page Rules can automatically configure custom domain support for COS buckets.

CNAME DNS records are created to alias the custom domain to the COS bucket hostname. All traffic to the custom domain will then be forwarded to the COS service.

When COS serves files from bucket sub-domains, the HTTP `Host` request header value to determine the bucket name. With CNAME DNS records, this header value will still refer to the custom domain, rather than the bucket sub-domain. This field

needs to be dynamically updated with the correct value.

Create IBM Cloud Internet Services instance

- Provision a new instance of Cloud Internet Services.

Register Custom Domain name with Cloud Internet Services

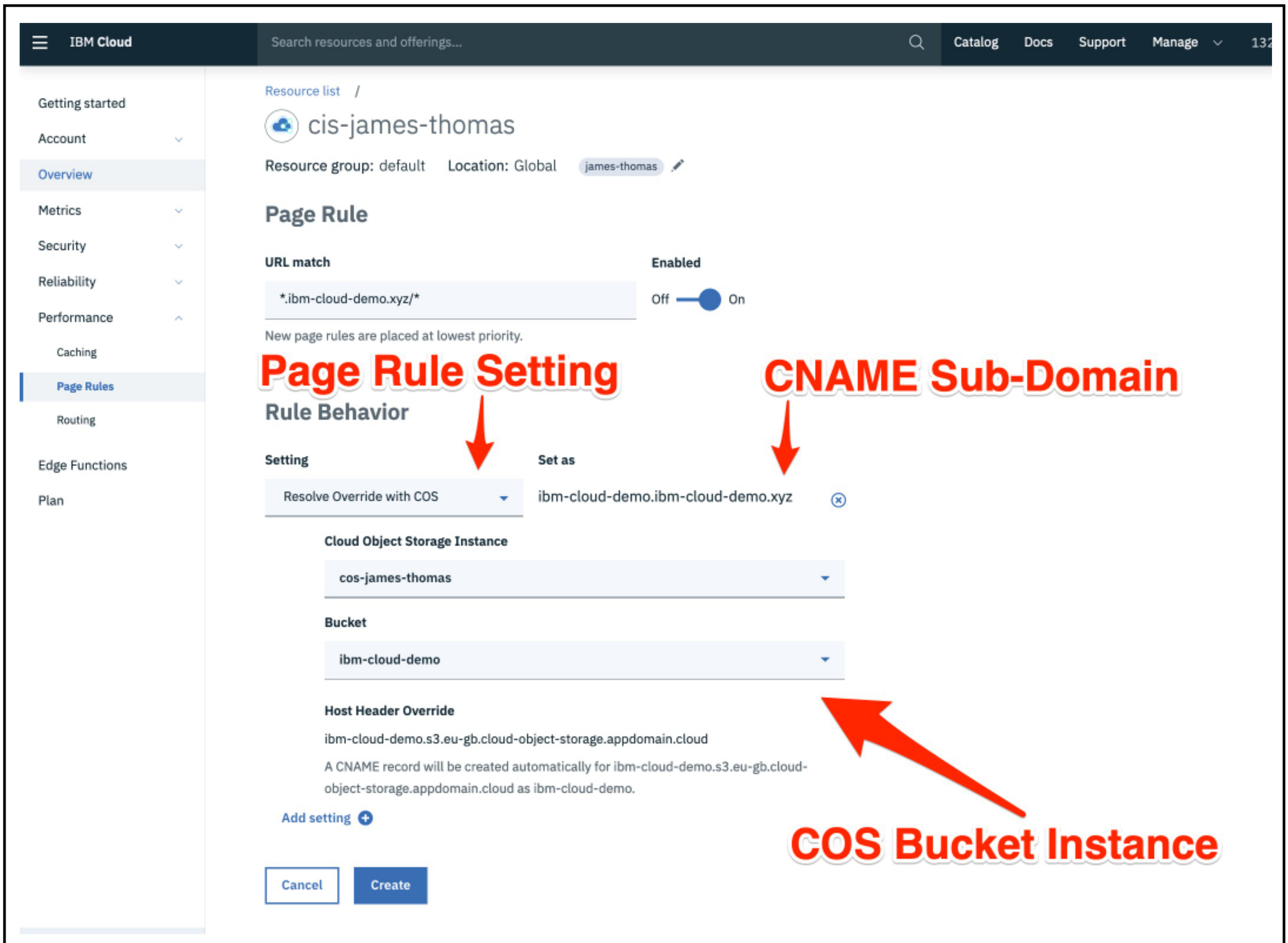
- Follow the documentation on how to register a custom domain with Cloud Internet Services.

This process involves delegating name server control for the domain over to IBM Cloud Internet Services.

Configure Page Rules and DNS records (automatic)

Cloud Internet Services can automatically set up Page Rules and DNS records needed to forward custom domain traffic to COS buckets. This automatically exposes the bucket as `bucket-name.your-domain.com`. If you want to change this default sub-domain name, follow the manual steps in the next section.

- Click the Performance drop-down menu and click the "*Page Rules*" link.
- Click the "*Create rule*" button from the table.
- Select the Rule Behaviour Setting as "*Resolve Override with COS*"
- Select the correct COS instance and bucket.
- Click the "*Create*" button.



Once DNS records have propagated, bucket files should be accessible using the custom domain: `http(s)://<CUSTOM_DOMAIN>/index.html`.

Configure Page Rules and DNS records (manual)

These steps only need following if you haven't done the section above….

Create the Page Rule to modify the HTTP host header.

- Click the Performance drop-down menu and select the "*Page Rules*" link.
- Click the "*Create rule*" button from the table.
- Set the URL match field to be `<SUB_DOMAIN>.<CUSTOM_DOMAIN>/*`
- Select the Rule Behaviour Setting as "*Host Header Override*" as the custom bucket sub-domain:`<BUCKET_NANME>.<REGION>.eu-gb.cloud-object-stor age.appdomain.cloud`

Create the DNS CNAME record to forward traffic to COS.

- Click the Reliability drop-down menu and click the "*DNS*" menu entry.
- Add a new DNS record with the following values.
    - **Type:** *CNAME*

    - **Name:** *<custom subdomain host>*
    - **TTL:** *Automatic*
    - **Alias Domain Name:** *<COS bucket sub-domain>*

Name is the sub-domain on the custom domain (e.g. www) through which the COS bucket will be accessible. *Alias Domain Name* is the COS bucket sub-domain from above, e.g. `<BUCKET_NANME>.<REGION>.eu-gb.cloud-object-storage.appdomain.cloud`

- Once the record is added, set the `Proxy` field to true. This is necessary for the page rules to work.

Once DNS records have propagated, bucket files should be accessible using the custom domain.

Configurable Index and Error pages

COS will now serve static assets from a custom sub-domain, where file names are explicitly included in the URL, e.g. `http(s)://<CUSTOM_DOMAIN>/index.html`. This works fine for static websites with two exceptions, the default document for the web site and the error page.

When a user visits the COS bucket sub-domain without an explicit file path (`http(s)://<CUSTOM_DOMAIN>`), the COS service will return the bucket file list, rather than the site index page. Additionally, if a user requests a missing file, COS returns an XML error message rather than a custom error page.

Both issues can be resolved using Edge Functions, a new feature in Cloud Internet Services.

Edge Functions

Edge functions are JavaScript source files deployed to Cloudflare’s Edge locations. They can dynamically modify HTTP traffic passing through Cloudflare’snetwork (for domains you control). Custom edge functions are triggered on configurable URL routes. Functions are passed the incoming HTTP request and control the HTTP response returned.

Add Edge Function to provide Index & Error Documents

Using a custom edge function, HTTP traffic to the custom sub-domain can be modified to support Index and Error documents. Incoming HTTP requests withoutan explicit file name can be changed to use the index page location. HTTP 404

responses returned from COS can be replaced with a custom error page.

- Open the "*Edge Functions*" page from the Cloud Internet Services instance homepage.
- Click the "*Create*" icon on the "*Actions*" tab.
- Enter "*route-index-and-errors*" in the action name field.
- Paste the following source code into the action body section.

*The **INDEX_DOCUMENT** and **ERROR_DOCUMENT** values control the index and error pages used to redirect requests. Replace these values with the correct page locations for the static site being hosted.*

```
const INDEX_DOCUMENT = 'index.html'
const ERROR_DOCUMENT = '404.html'

addEventListener('fetch', event => {
  event.respondWith(handleRequest(event.request))
})

async function handleRequest(request) {
  const url = new URL(request.url)

  // if request is a directory path, append the index
document.
  if (url.pathname.endsWith('/')) {
    url.pathname = `${url.pathname}${INDEX_DOCUMENT}`
    request = new Request(url, request)
  }

  let response = await fetch(request)

  // if bucket file is missing, return error page.
  if (response.status === 404) {
    url.pathname = ERROR_DOCUMENT
    request = new Request(url, request)
    response = await fetch(request)

    response = new Response(response.body, {
      status: 404,
      statusText: 'Not Found',
      headers: response.headers
```

```
    })
  }

  return response
}
```

- Click the "*Save*" button.

Set up Triggers for Edge Function

- Select the "*Triggers*" panel from the Edge Functions page.
- Click the "*Add trigger*" icon.
- Set the Trigger URL to `http://<SUB_DOMAIN>.<CUSTOM_DOMAIN>/*`.
- Select the "*route-index-and-errors*" action from the drop-down menu.
- Click the "*Save*" button.

Test Index and Error Pages

Having set up the trigger and edge function, HTTP requests to the root path on the custom sub-domain will return the index page. Accessing invalid bucket files willalso return the error page, rather than the COS error response.

- Confirm that `http://<SUB_DOMAIN>.<CUSTOM_DOMAIN>/` returns the same page as `http://<SUB_DOMAIN>.<CUSTOM_DOMAIN>/index.html`
- Confirm that `http://<SUB_DOMAIN>.<CUSTOM_DOMAIN>/missing-page.html` returns the error page. This should be different to the XML error response returned by visiting `<BUCKET_NANME>.s3.<REGION>.cloud-object-storage.appdomain.cloud/missing-page.html`.

If this all works - the site is working! IBM Cloud is now hosting a static websiteusing Cloud Object Storage and Cloud Internet Services with Page Rules and Edge Functions.

Summary

Static web sites can be hosted on IBM Cloud using Cloud Object Storage andCloud Internet Services.

Cloud Object stores page files needed to render the static website. Anonymous bucket file access means files are accessible as public HTTP endpoints, without

having to run infrastructure to serve the assets.

Cloud Internet Services forwards HTTP traffic from a custom domain to the bucket hostname. DNS CNAME records are used to resolve the sub-domain as thecustom bucket hostname. Page Rules override HTTP request headers to make this work. Edge Functions are used to implement configurable Index and Error documents, by dynamically modifying in-flight requests with custom JavaScript.

Hosting static web sites using this method can be much cheaper (and easier) than traditional infrastructure. Developers only get charged for actual site usage,based on bandwidth and HTTP requests.

So thus our blog has been deployed in the internet using our IBM cloud services.