CloudFront Tasks

1) Configure VPC peering in cross regions.

Step 1: Create VPC Peering Connection

* Go to **VPC Dashboard** in **Region A (us-north-1)**.
* In the left menu, click **"Peering Connections"** → Click **"Create Peering Connection"**.
* Fill in the details:
* **Name tag**: cross-region-peer
* **VPC Requester**: Select **VPC-A**
* **Account**: My account (assuming same AWS account)
* **Region**: Select **us-east-2**
* **VPC Accepter**: Select **VPC-B** from dropdown
* Click **Create Peering Connection**

Step 2: Accept the Peering Request

1. Switch to Region B (**us-east-2**).
2. Go to **VPC Dashboard** → **Peering Connections**
3. Select the pending connection → Click **"Actions"** → **Accept Request**

Now the VPCs are peered, but they **cannot talk yet** without route and security settings.

Step 3: Update Route Tables

**In Region A (us-north-1):**

1. Go to **Route Tables**
2. Find the route table for **VPC-A**
3. Click **"Routes"** tab → **Edit Routes** → **Add Route**
   * **Destination**: 172.168.0.0/16 (CIDR of VPC-B)
   * **Target**: Select the **Peering Connection (pcx-xxxx)**
4. Click **Save changes**

**In Region B (us-east-2):**

1. Repeat the same steps for **VPC-B’s** route table:
   * Destination: 192.128.0.0/24
   * Target: **Peering Connection**

**Step 4: Update Security Groups**

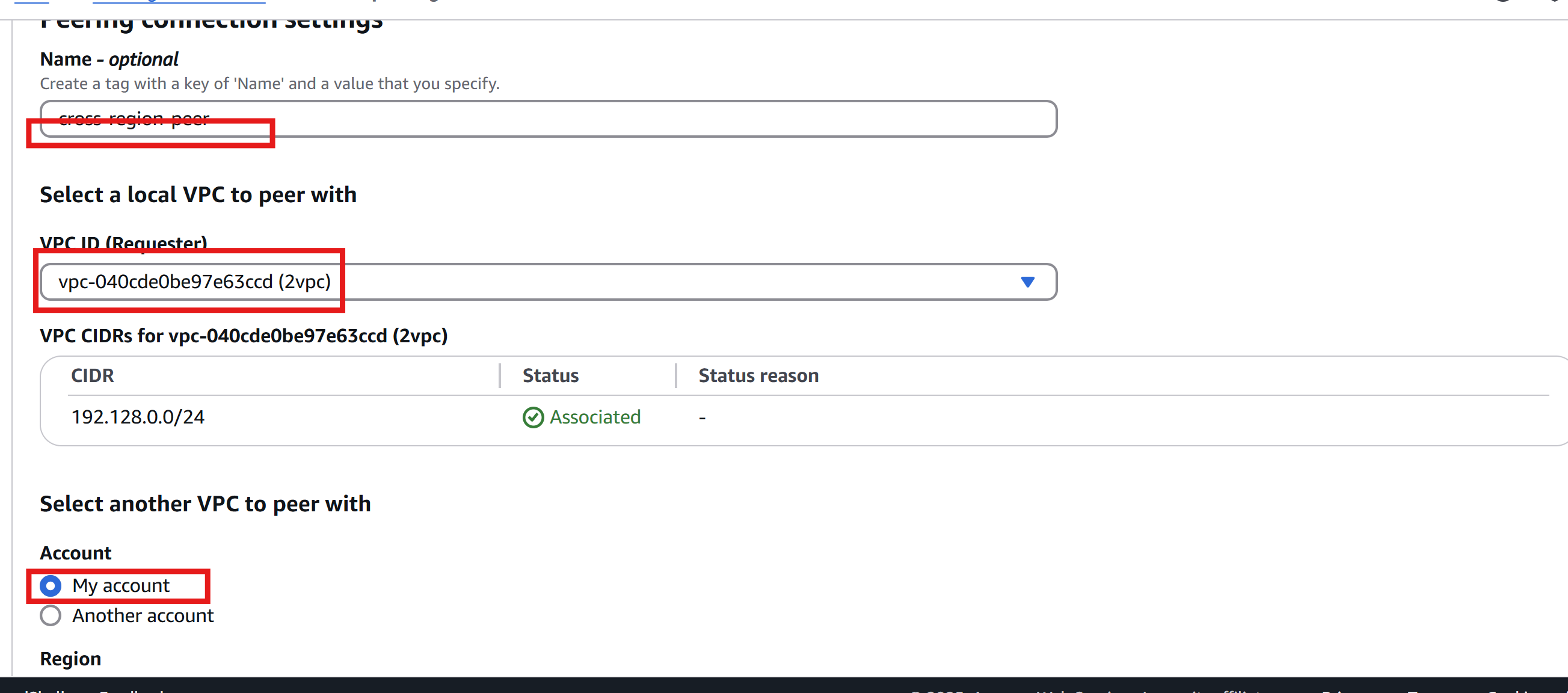
You need to allow traffic **from the other VPC’s CIDR**.

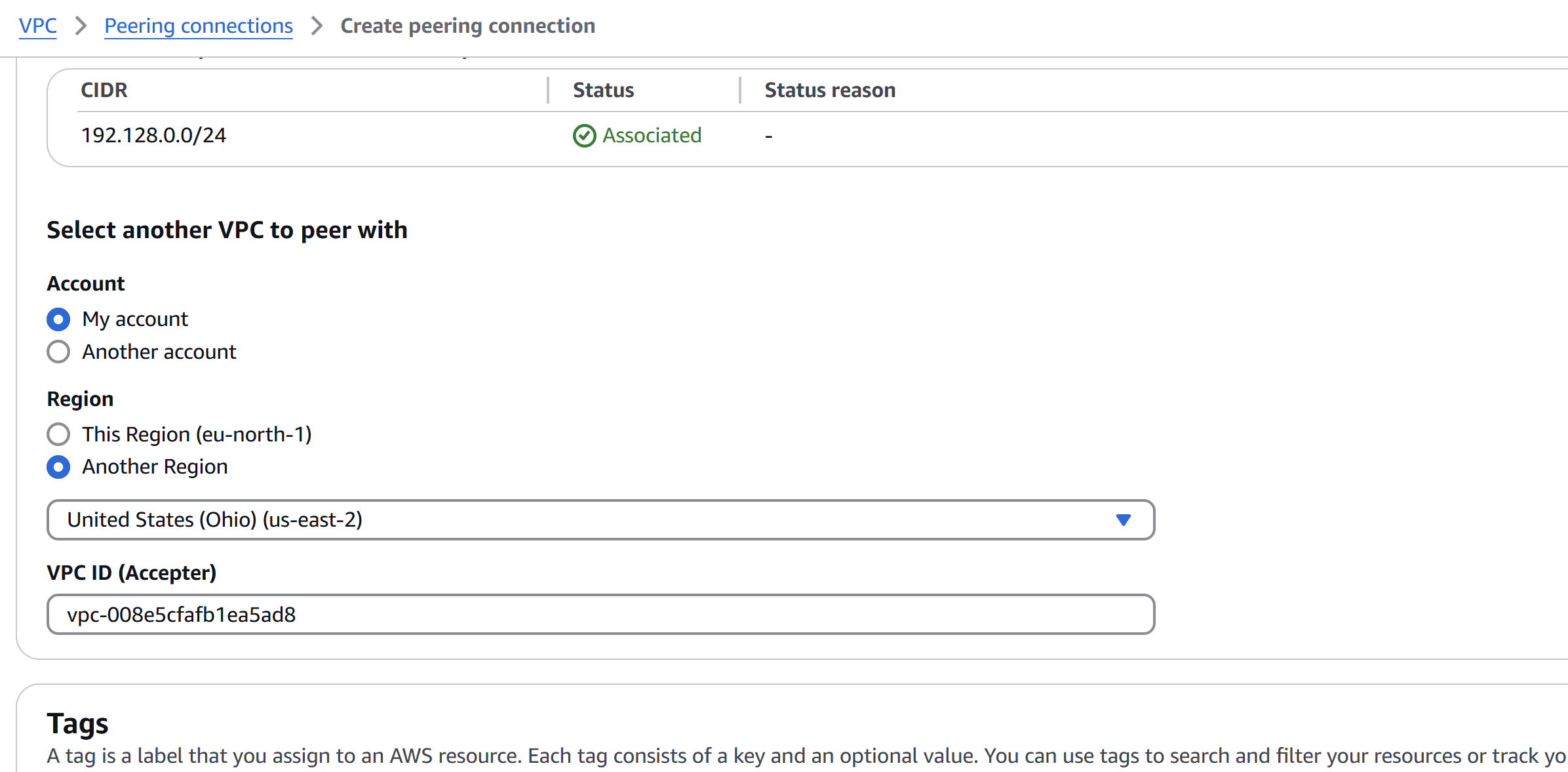
**In VPC-A:**

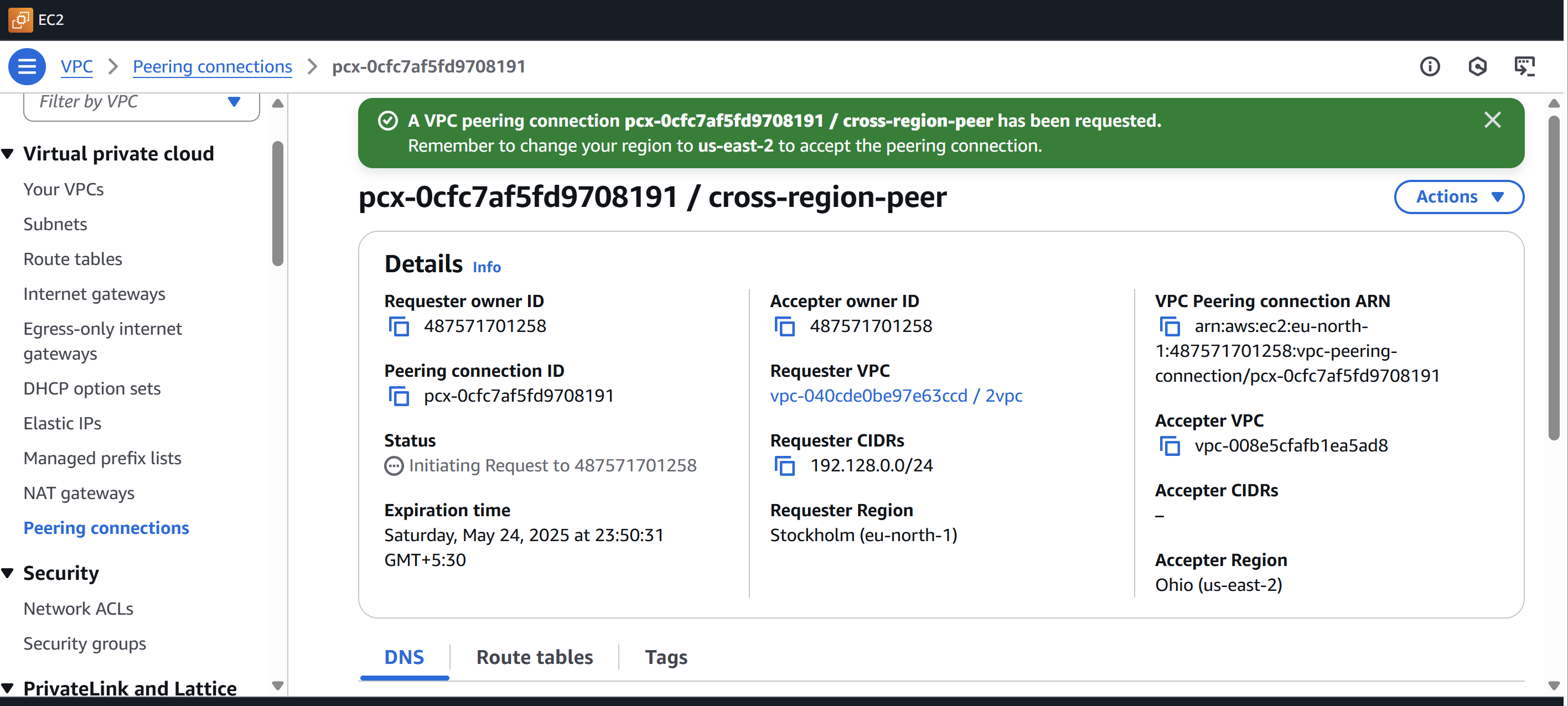
1. Go to **EC2** → **Security Groups**
2. Select the group attached to your instance
3. Click **Inbound rules** → **Edit**
4. Add rule:
   * Type: All traffic (or specific ports like SSH, HTTP)
   * Source: 172.168.0.0/16

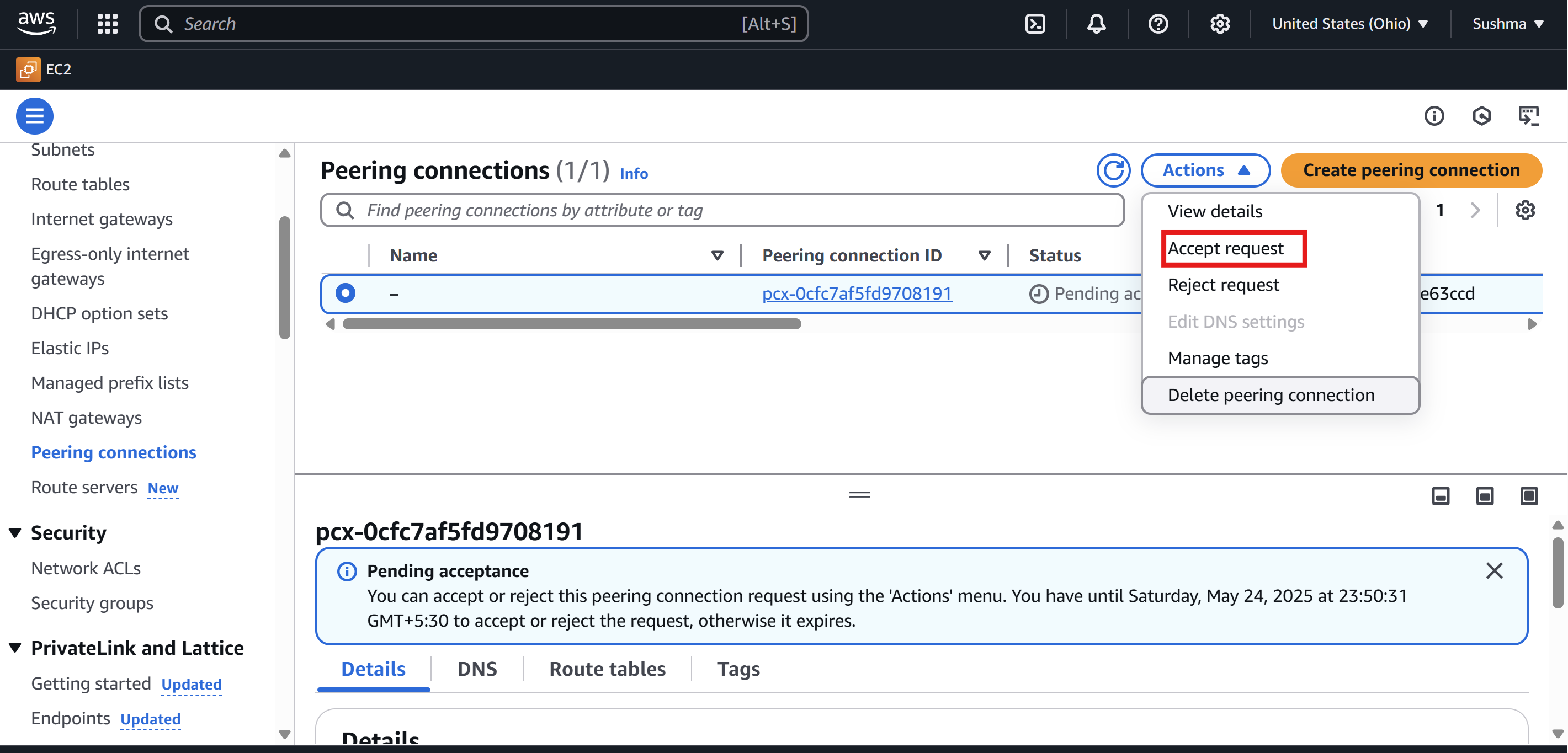
**In VPC-B:**

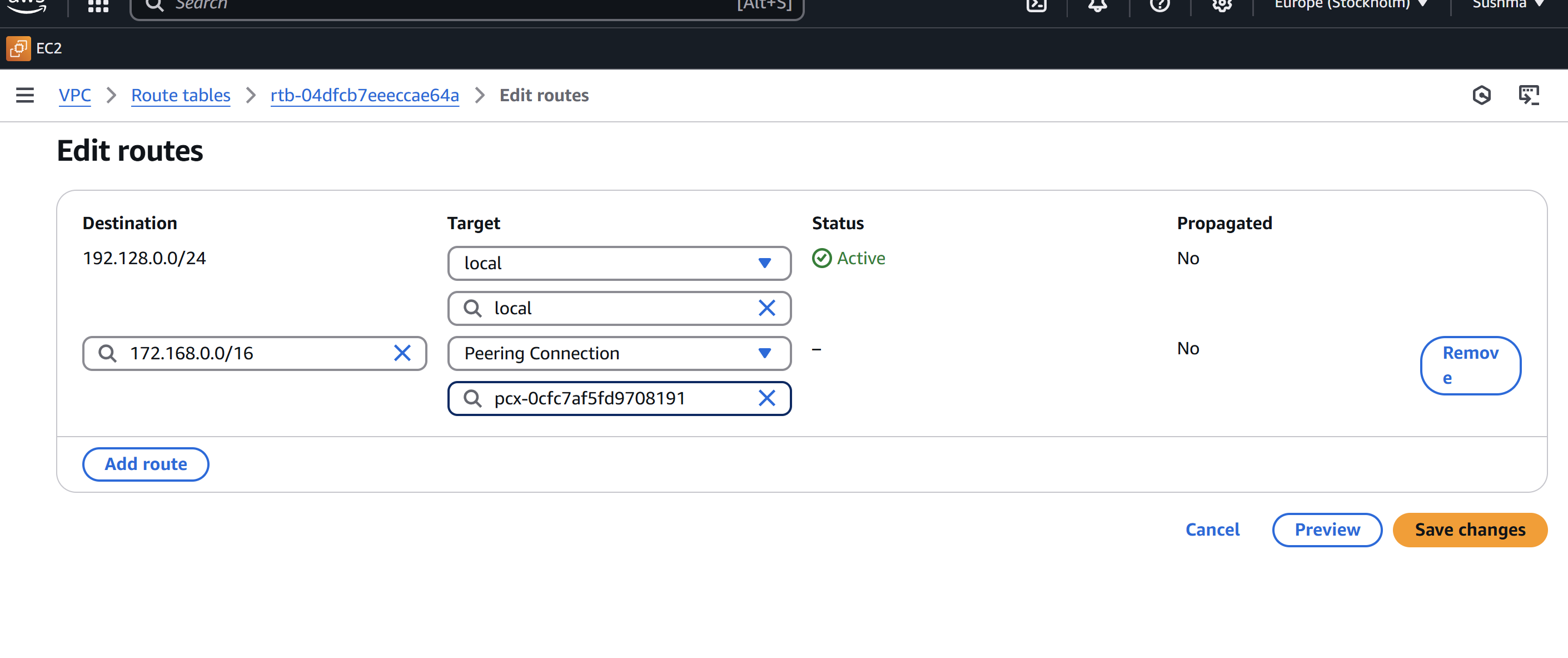
Repeat with 192.128.0.0/24 as the source.

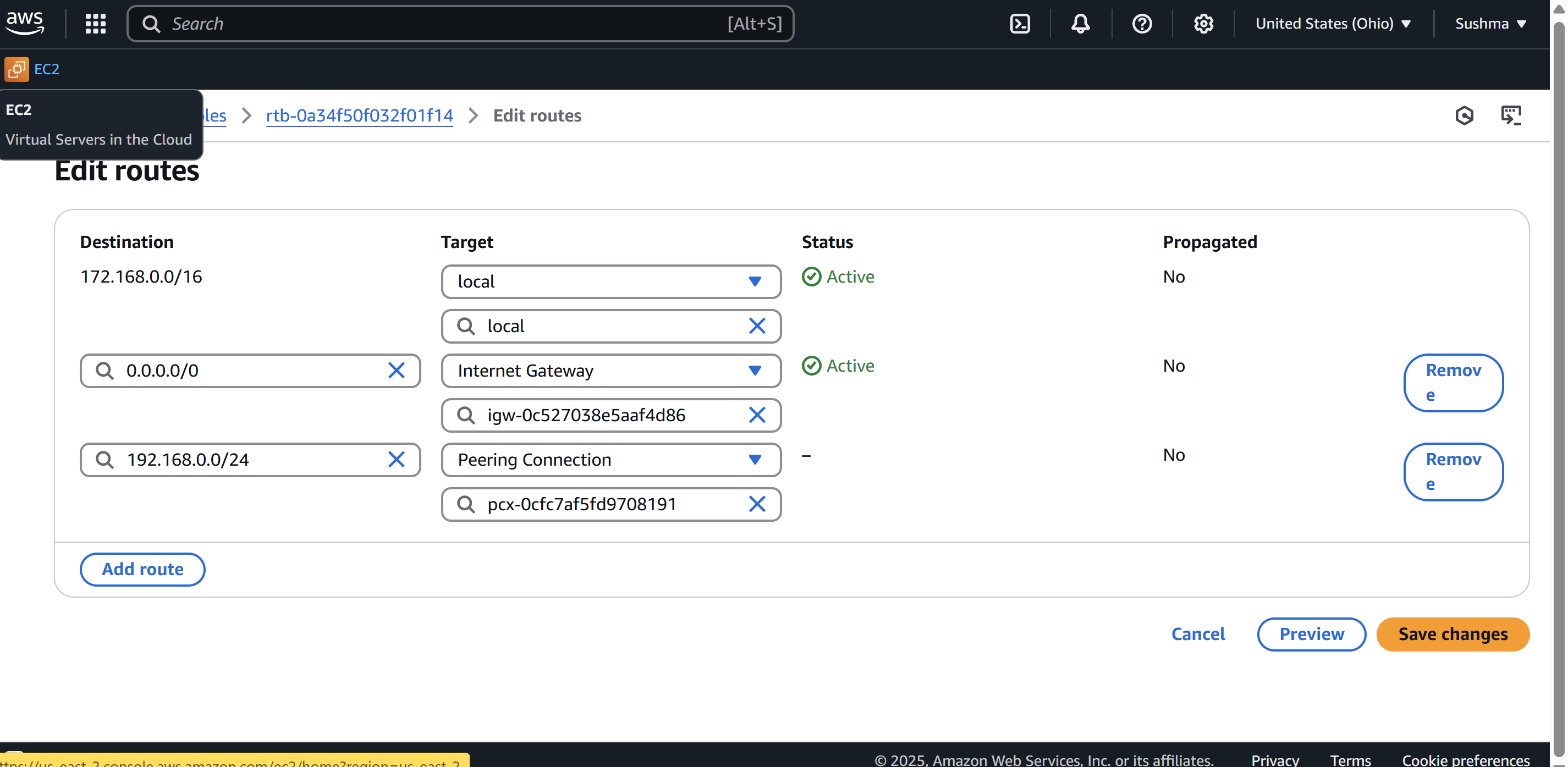


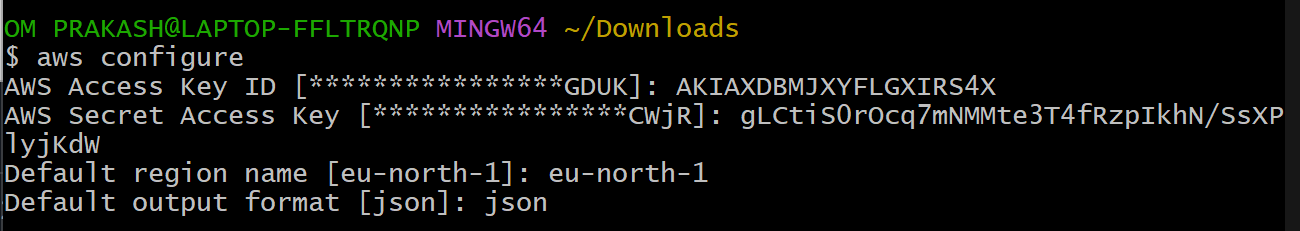


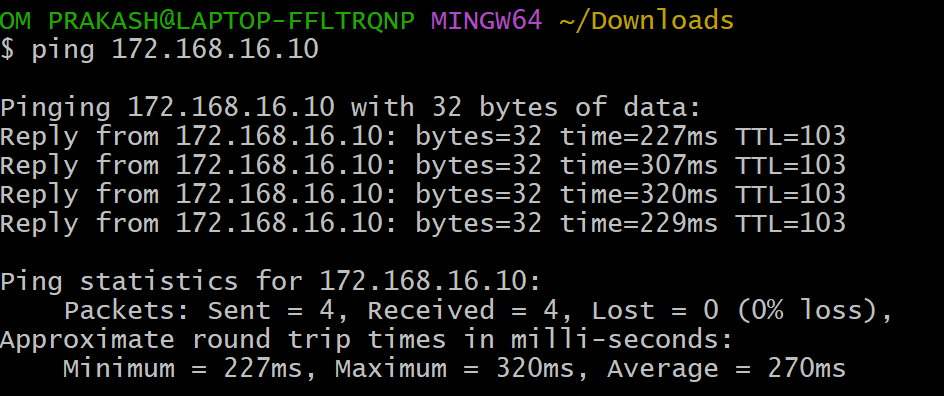




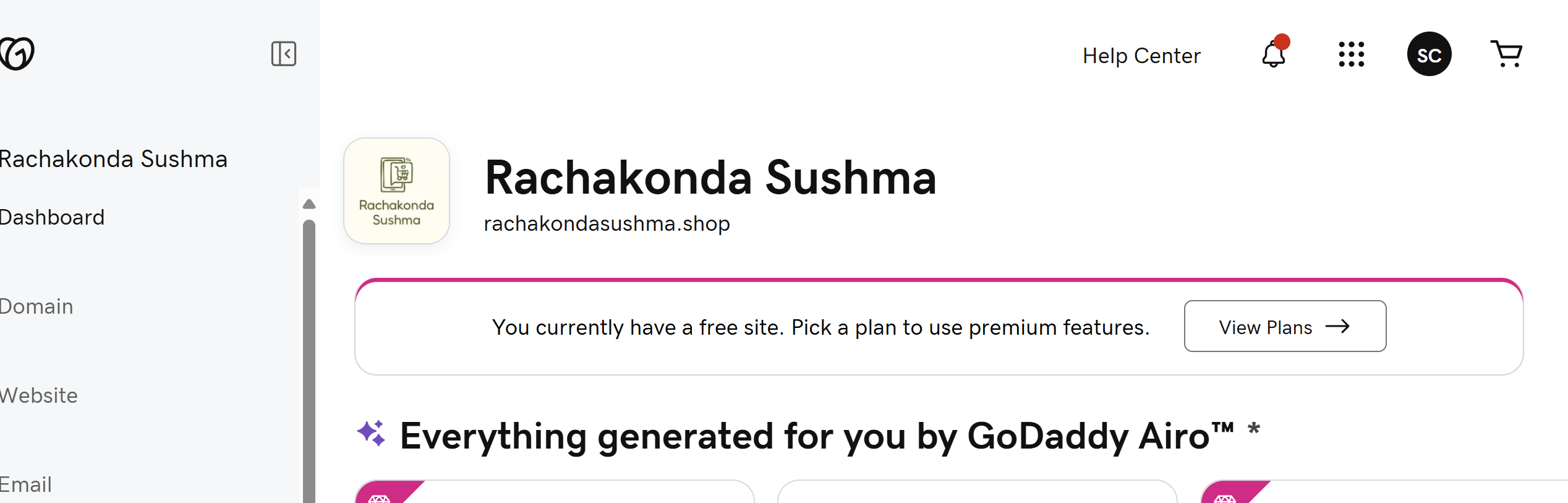








2) Purchase one domain from godaddy.

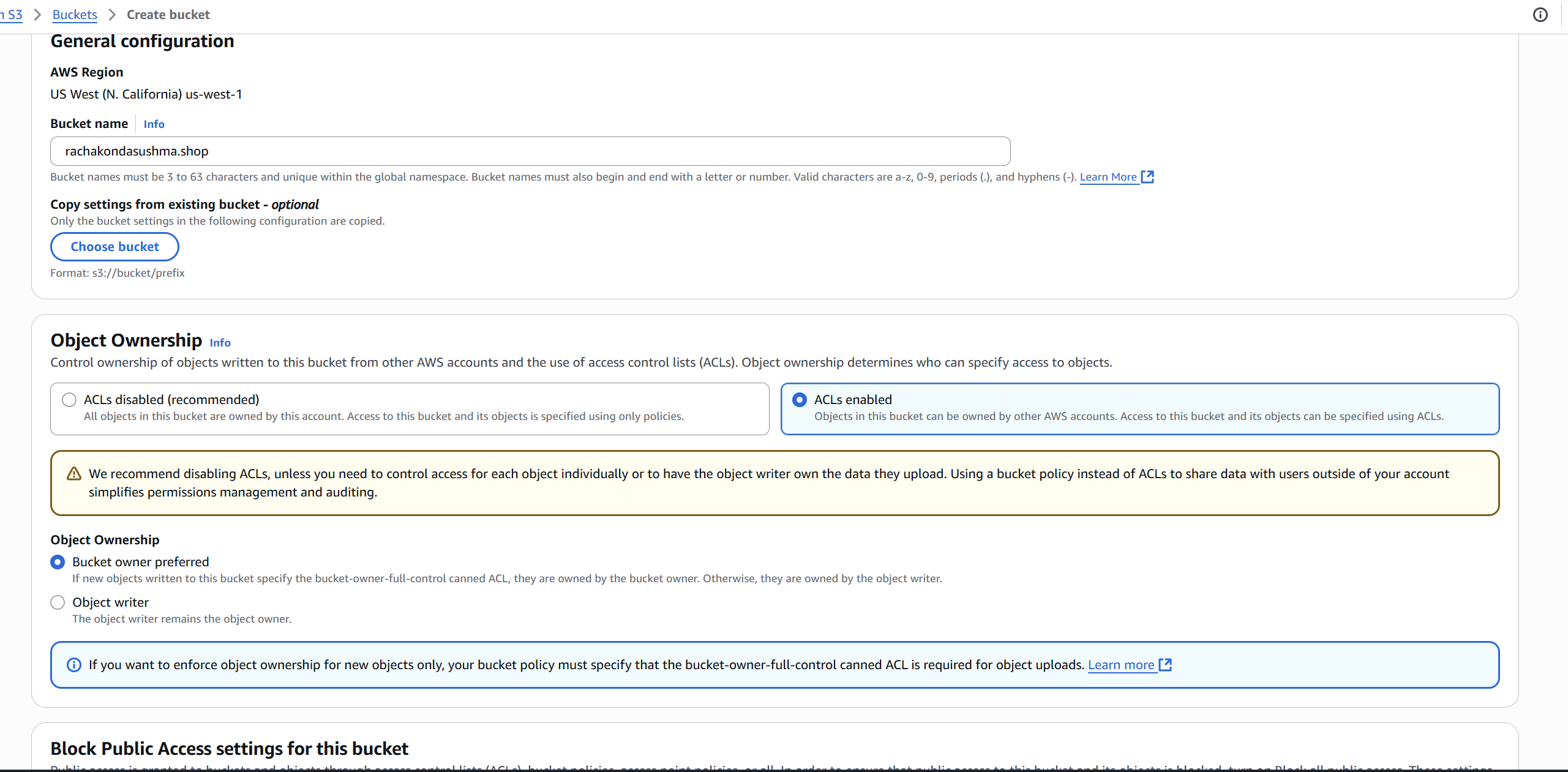


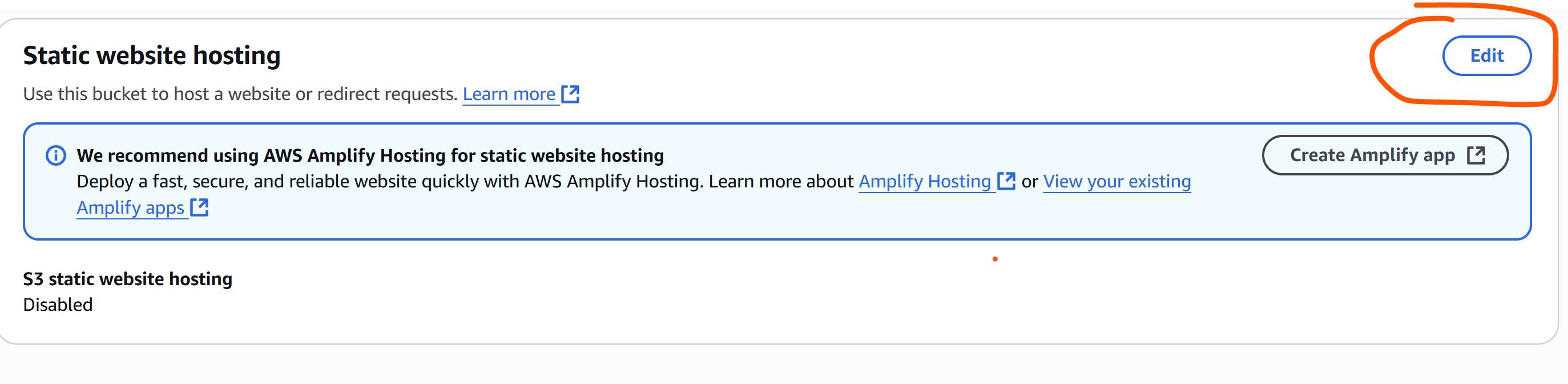
3) Deploy static webiste in s3.

* First we need to create two files here I am created as **index.html** and **error.html**

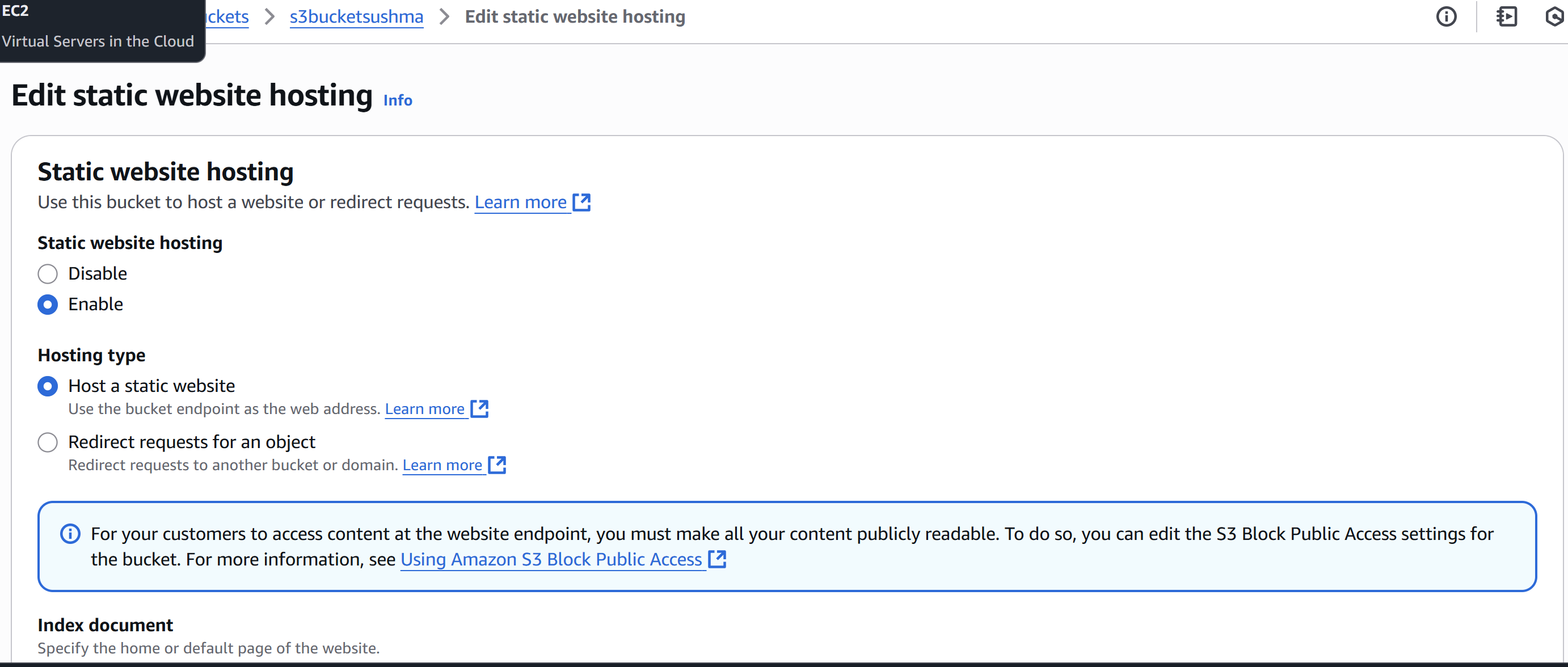
Which are used to get the page and error page.

* Then now go to **Buckets 🡪 select the bucket**

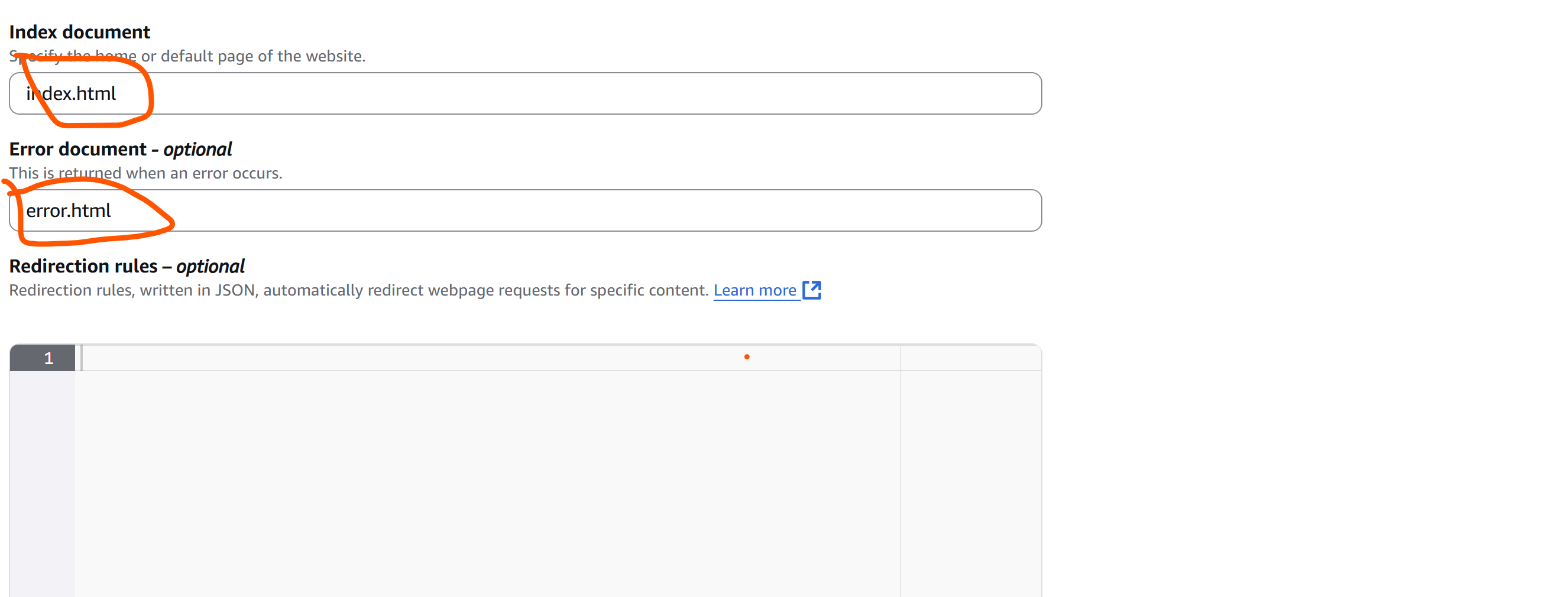




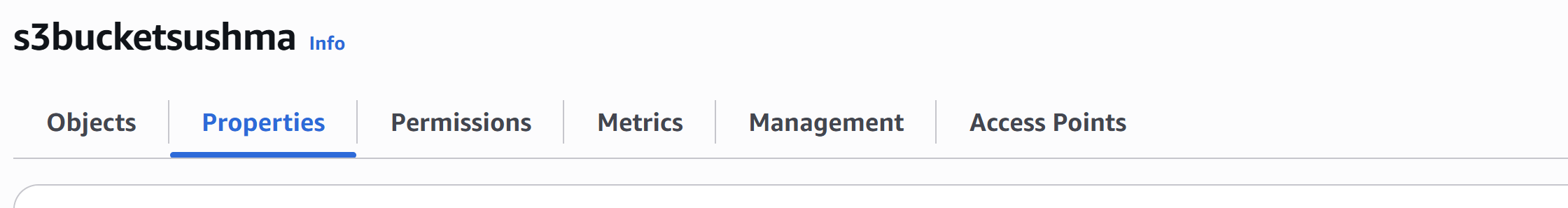
* Now go to  **properties -🡪 enable the static website hosting.**
* Select the Hosting type here I have selected **Static**

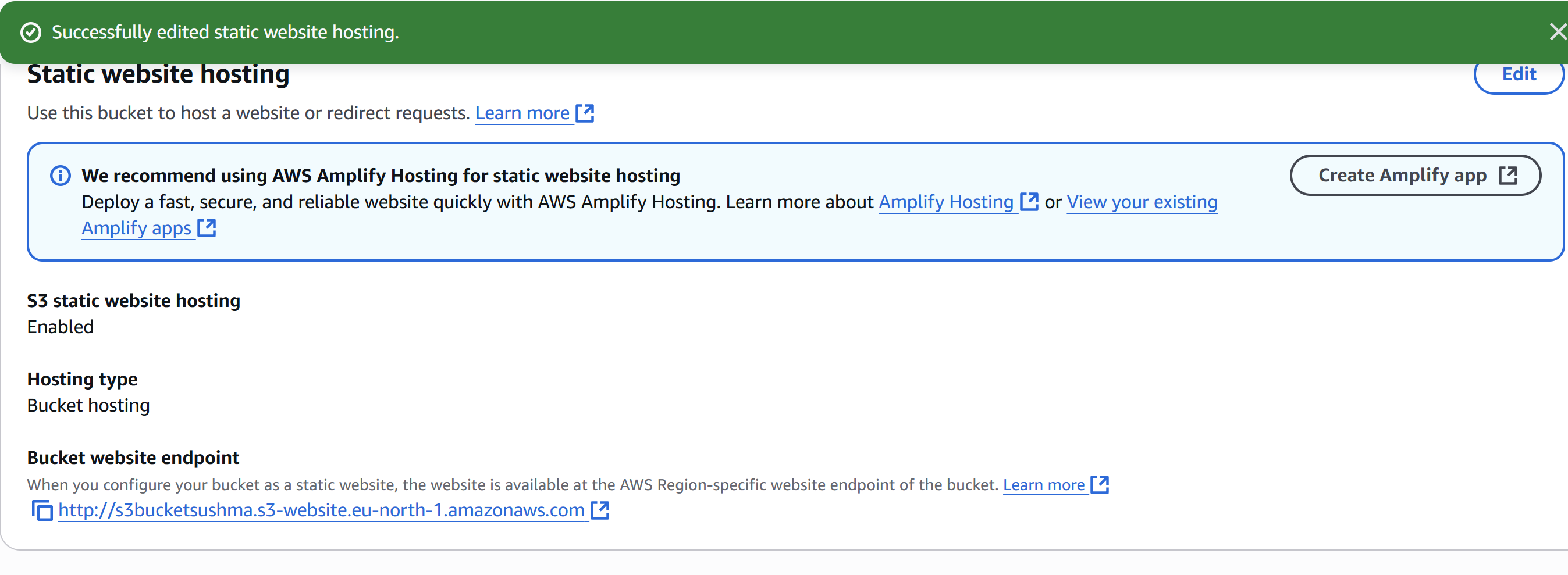


* Now enter the index object name which we uploaded in bucket **index.html and error.html** then save.

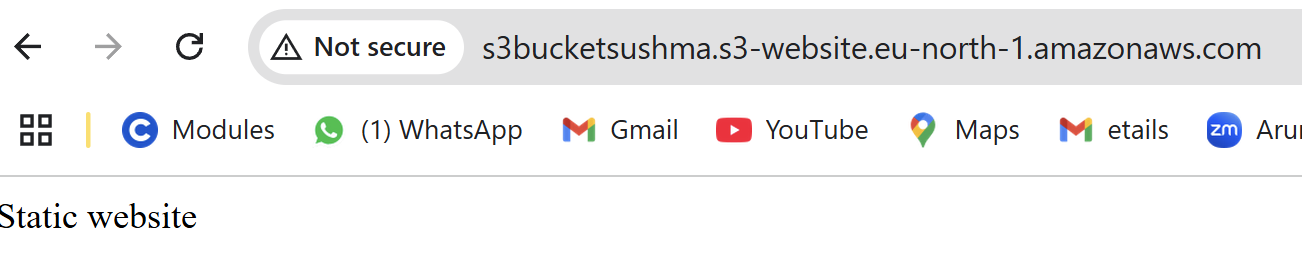


* Then go to the **bucket website endpoint**  and copy the link.

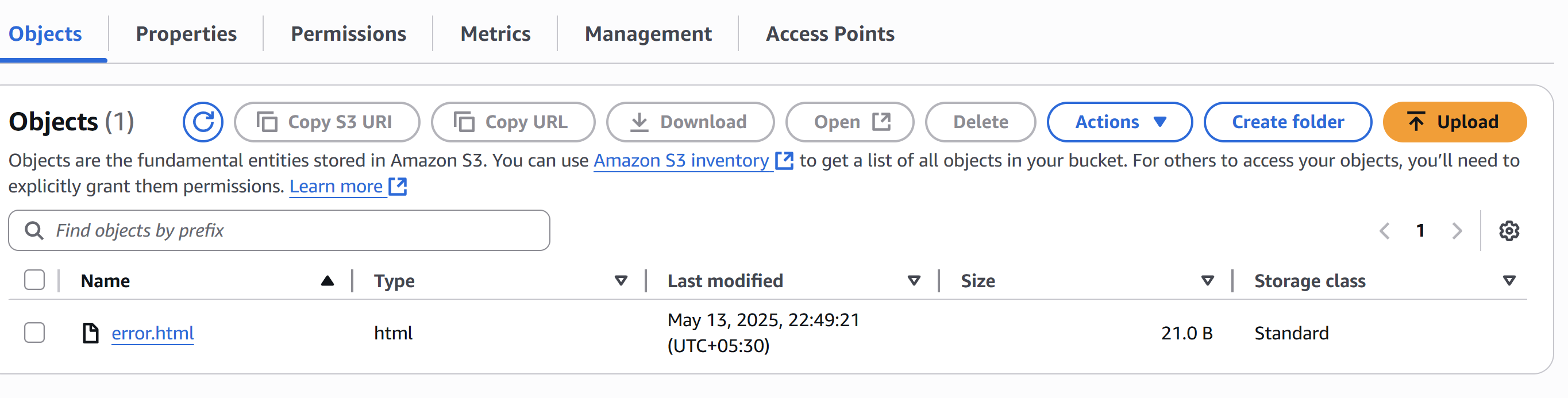




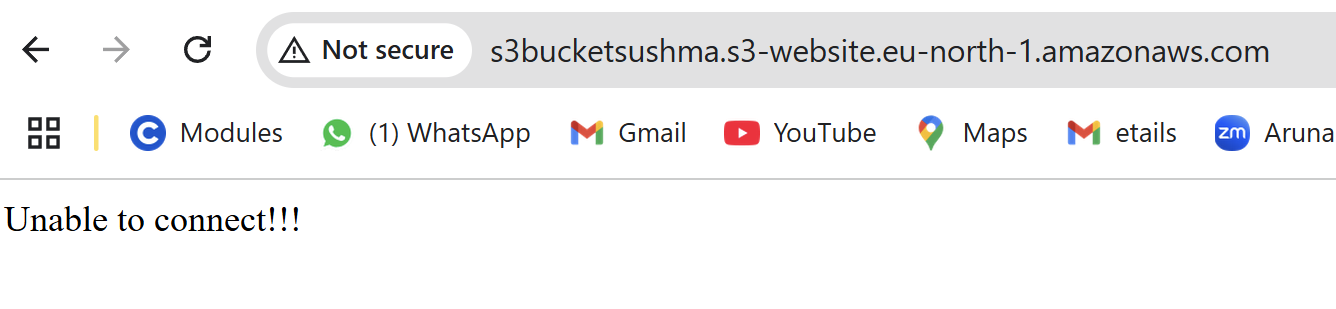
* Then deploy in the url then we will get the **Page with text which is in index.html**



* Now once delete the index object in bucket



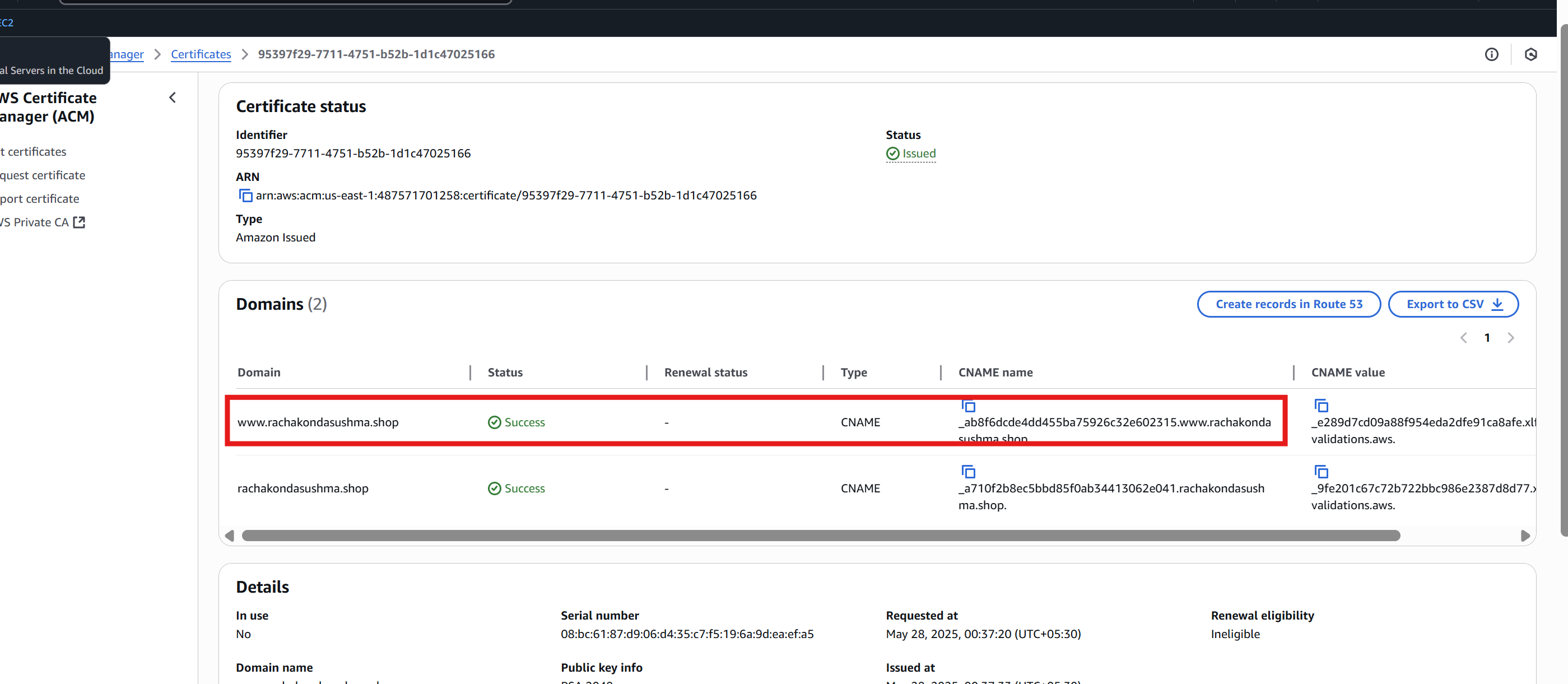
* Then try to reload the page then we will get the error message which is in error.html object.



4) Create CDN and attach one SSL certificate.

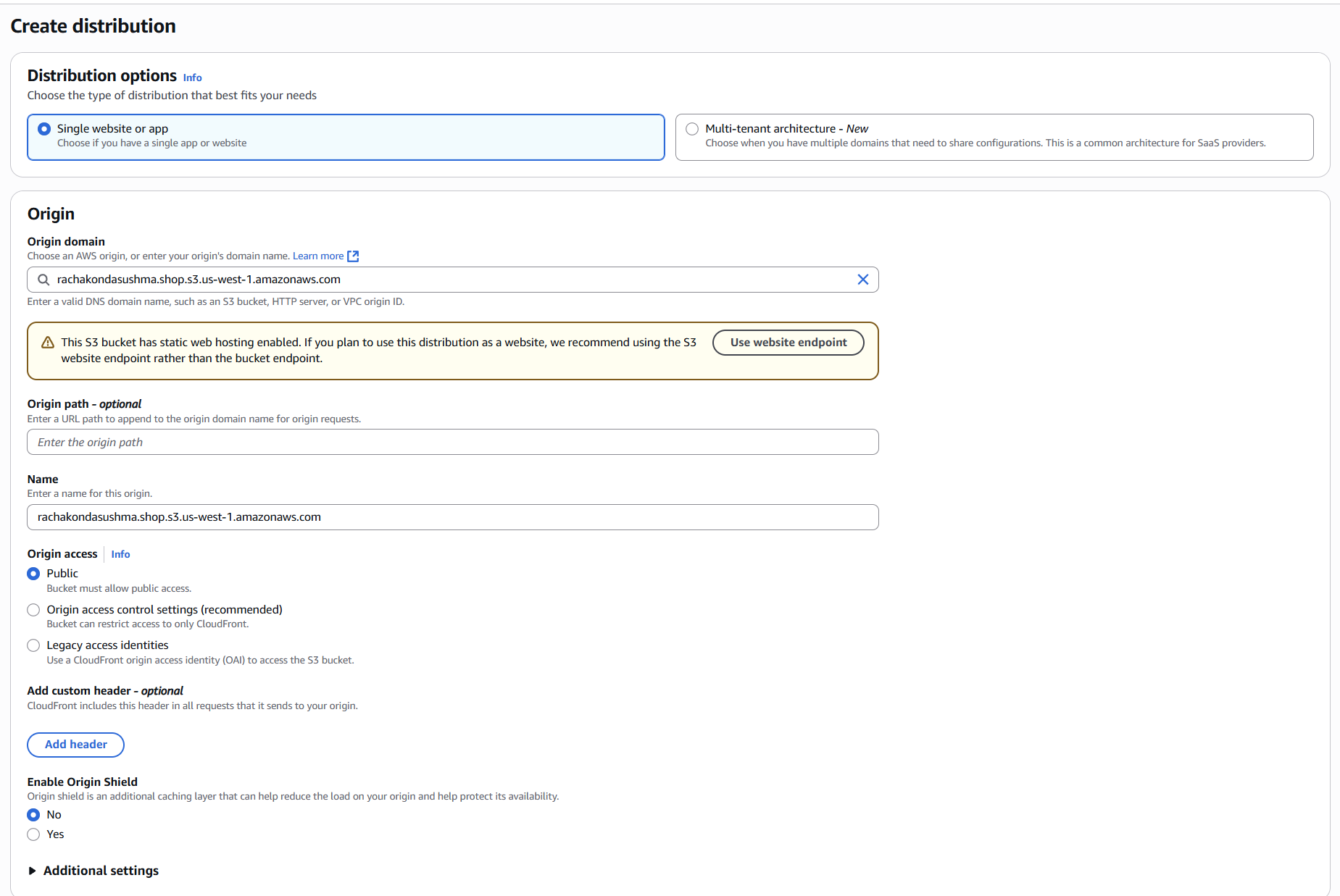
Step 1: Request SSL Certificate (ACM)

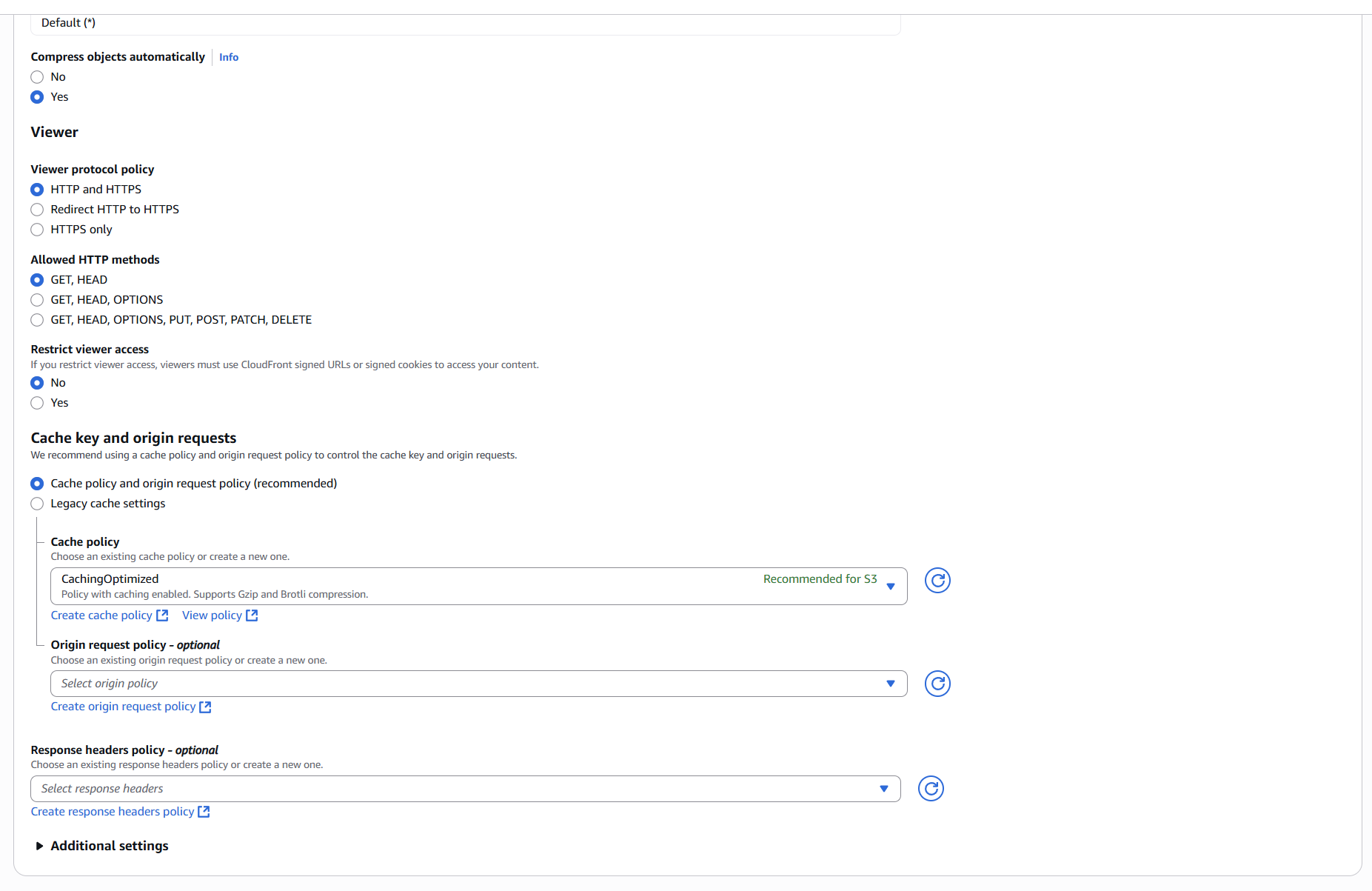
* Open **AWS Certificate Manager (ACM)**: <https://console.aws.amazon.com/acm/home?region=us-east-1>
* Click **“Request a certificate”**.
* Choose **Public certificate** → click **Next**.
* Enter your domain name (e.g., cdn.example.com) → click **Next**.
* Choose **DNS validation** (recommended) → click **Next**.
* Review and submit the request.
* Copy the DNS record from the validation step and add it to your DNS provider (e.g., Route 53 or GoDaddy).
* Wait until the status becomes **“Issued”**.

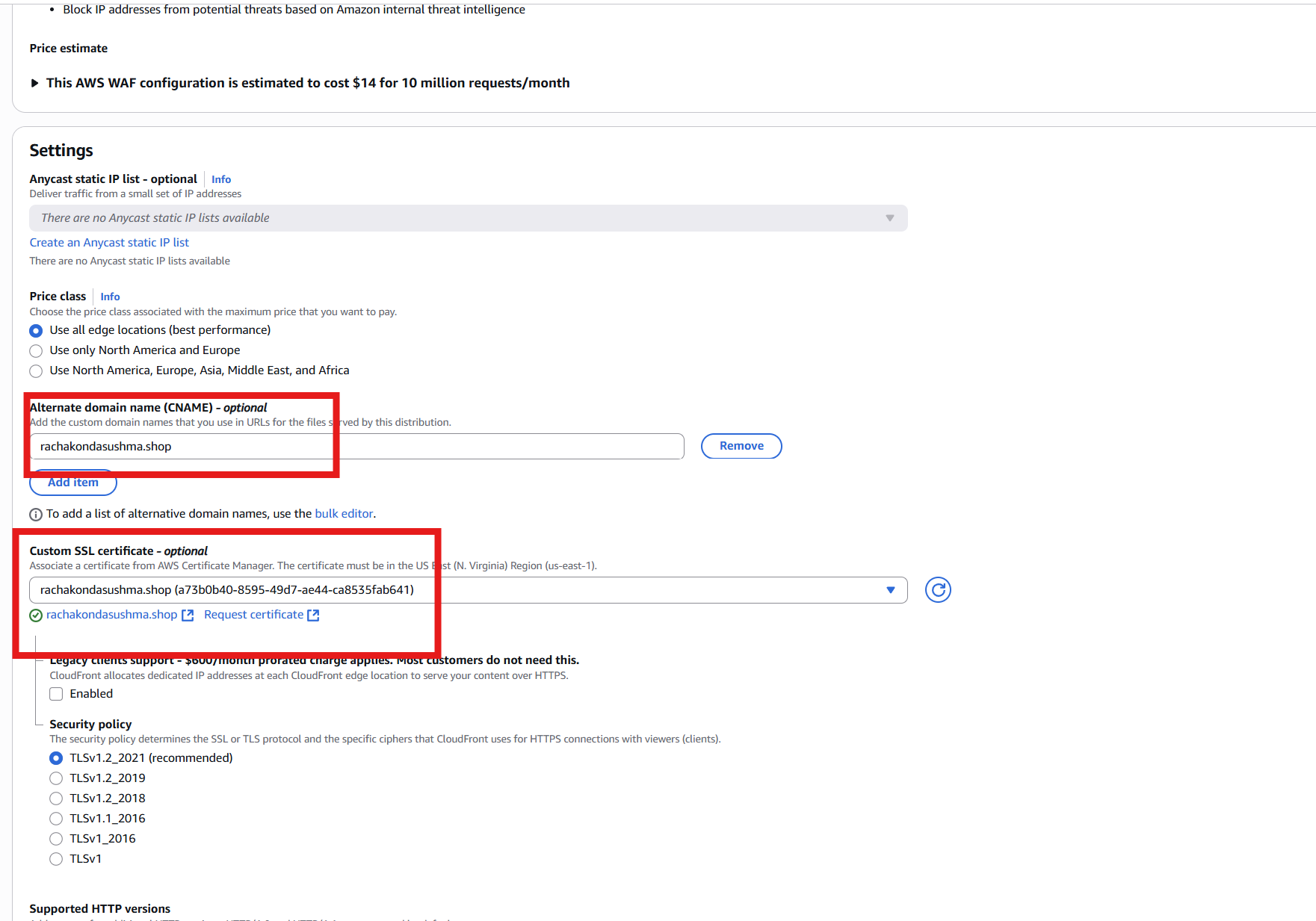


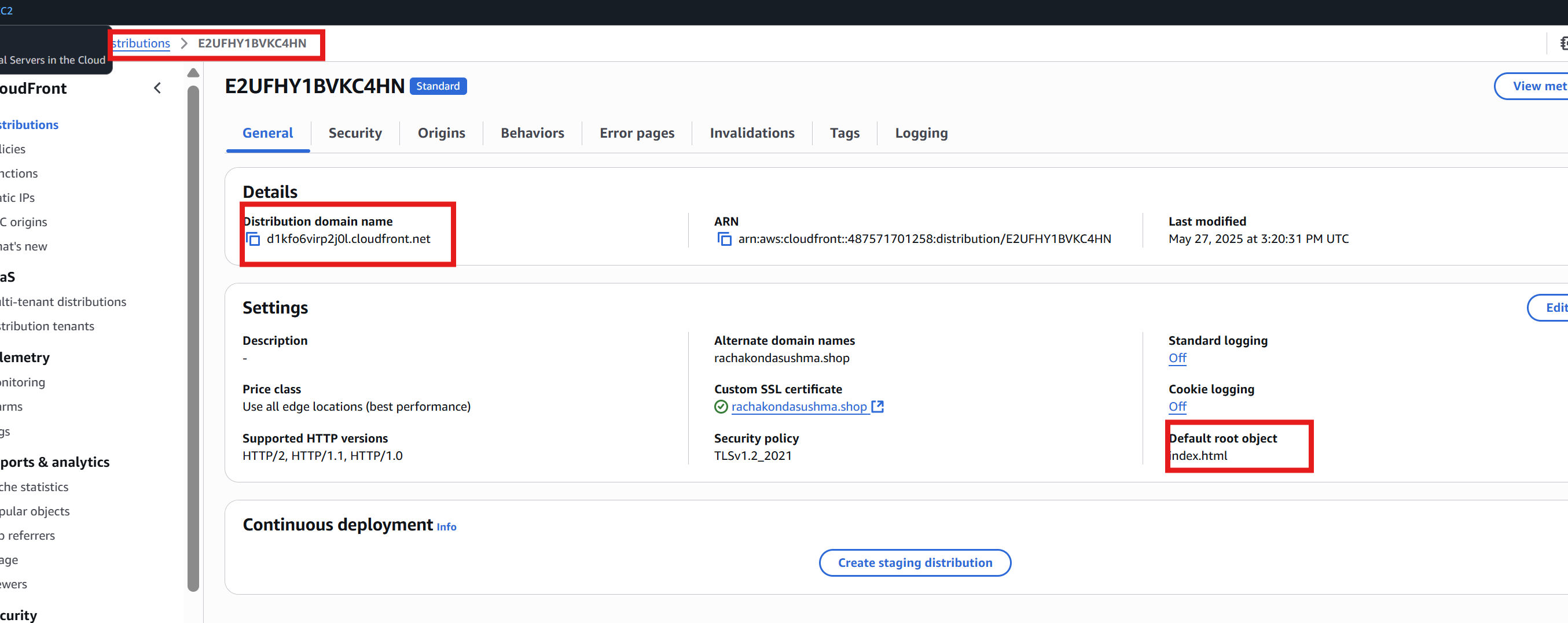
Step 2: Create CloudFront Distribution

* Open **CloudFront Console**: <https://console.aws.amazon.com/cloudfront>
* Click **“Create distribution”**.
* Under **Origin**:
  + - Origin domain: choose your S3 bucket or custom origin.
    - Name: auto-filled.
* Under **Default cache behavior**:
  + - Viewer protocol policy: select **Redirect HTTP to HTTPS**.
* Under **Settings**:
* Alternate domain name (CNAME): enter cdn.example.com.
* Select **Custom SSL certificate** → choose the ACM certificate you requested earlier.
* Click **Create distribution**.



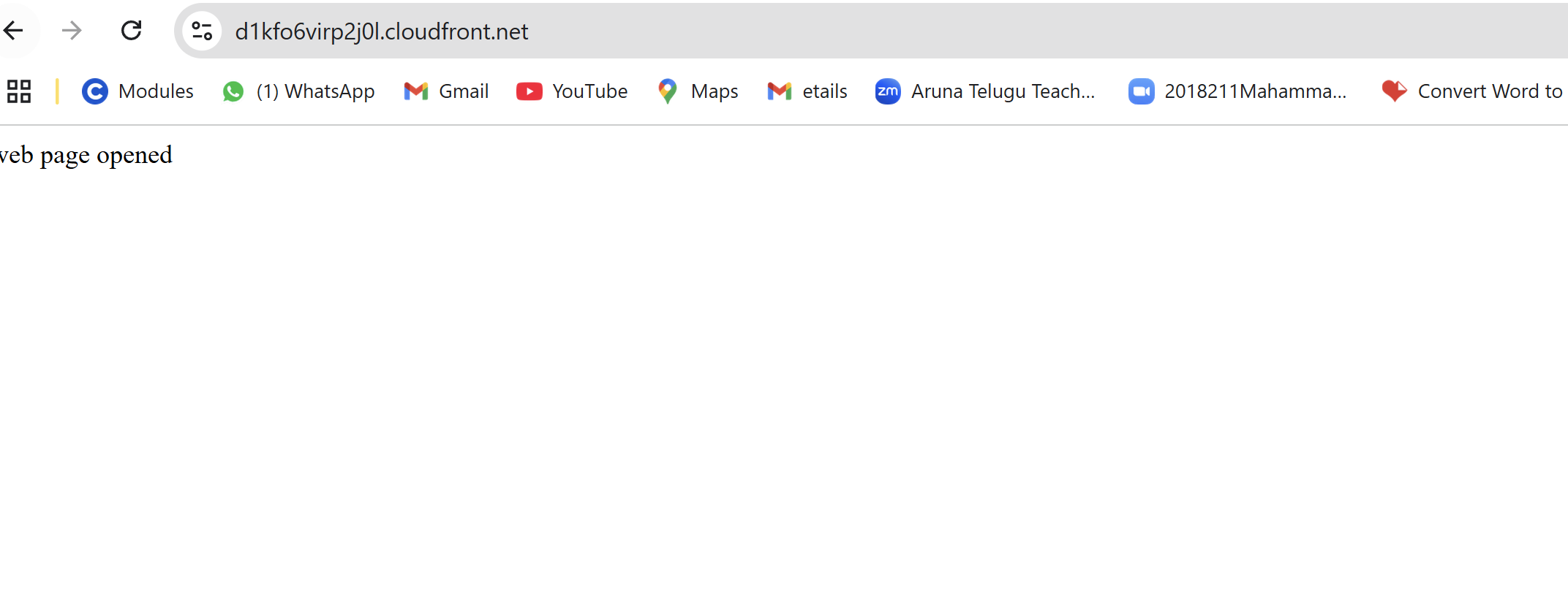






Step 3: Configure DNS for Custom Domain

* Go to your DNS provider (e.g., Route 53, GoDaddy).
* Create a **CNAME record**:
  + Name: rachakondasushma.shop
  + Value: your CloudFront domain (e.g., d1234abcd.cloudfront.net)

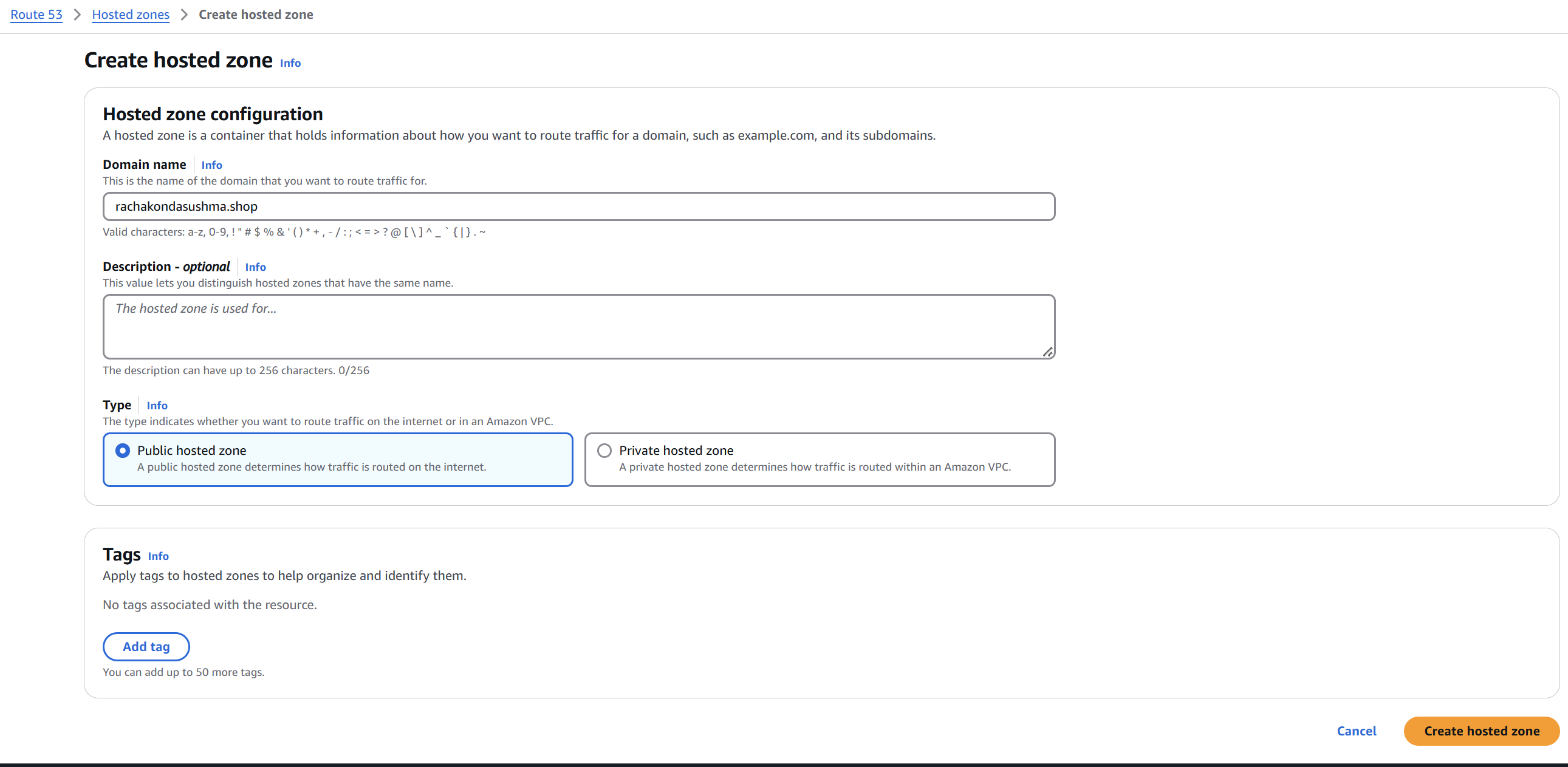


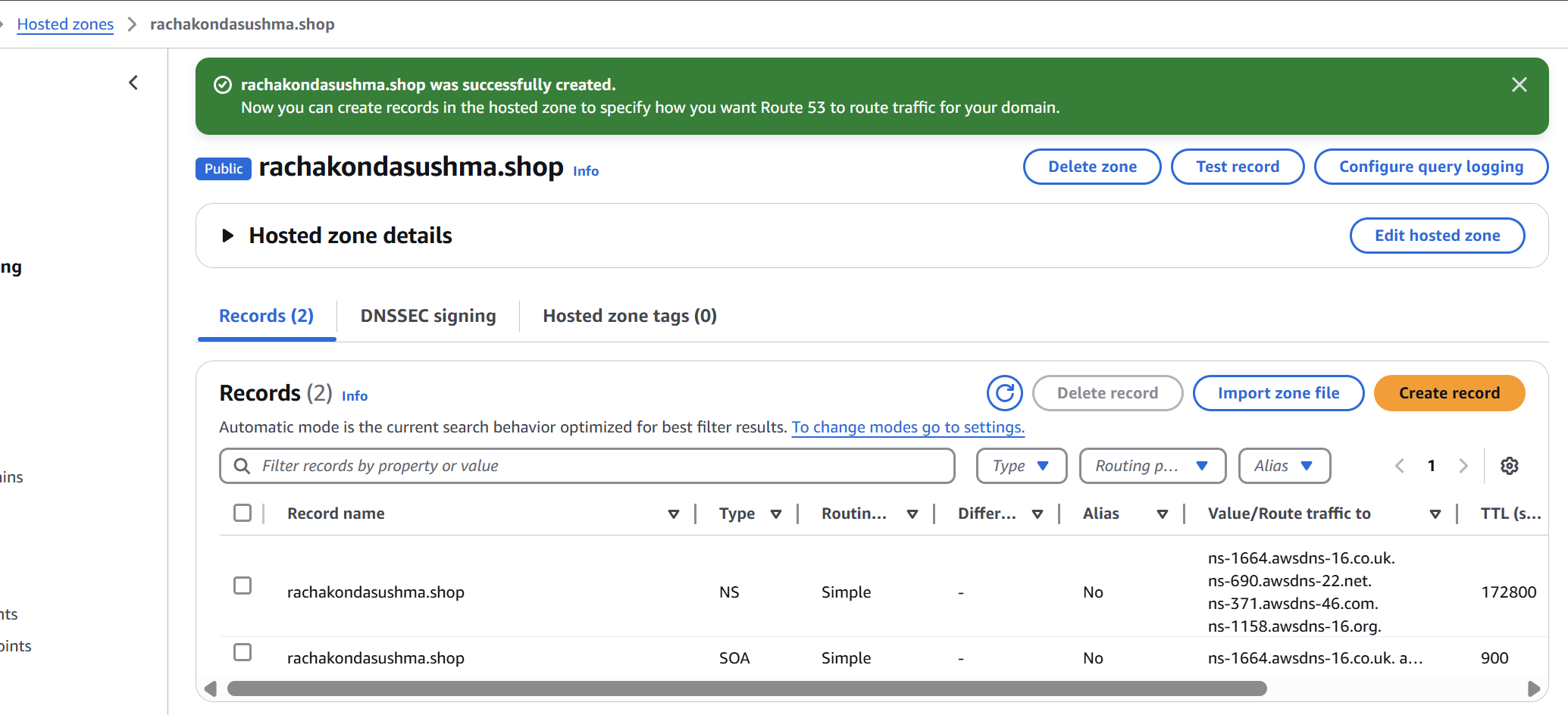
5) Create Route53 hosted zone and MAP the domain with CDN.

Step 1: Create a Public Hosted Zone in Route 53

* + **Go to the AWS Route 53 Console**
  + Click on **"Hosted zones"** → **"Create hosted zone"**
  + **Enter domain name** (e.g., example.com)
  + **Type**: Choose **Public Hosted Zone**
  + Click **"Create hosted zone"**

After creation, you’ll see **NS (Name Server)** and **SOA** records generated automatically.

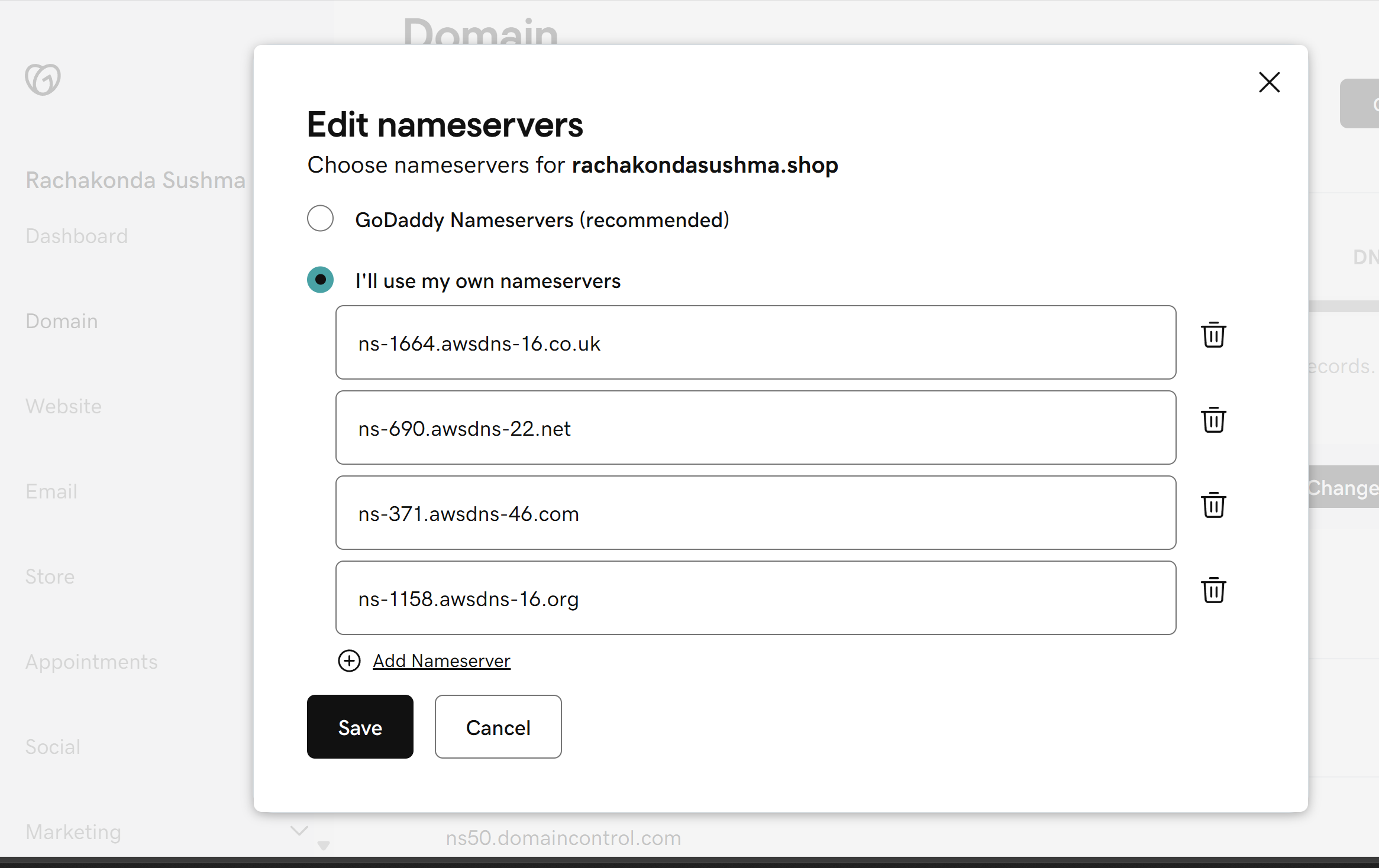




Step 2: Point Domain to AWS Name Servers

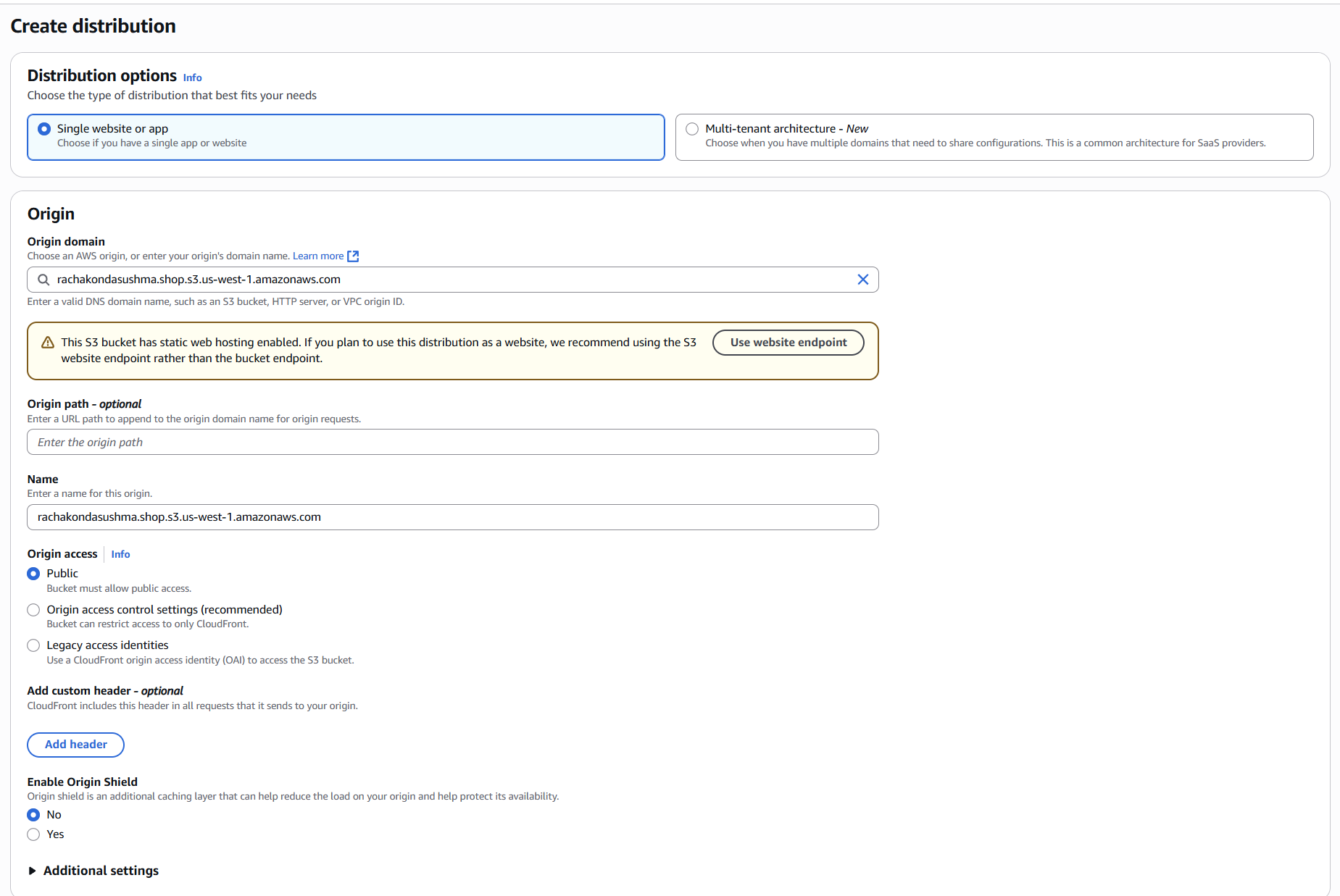
We need to update our domain registrar (e.g., GoDaddy, Namecheap) with the **NS records** from Route 53.

* Copy the **NS records** from Route 53.
* Go to your **domain registrar's DNS settings**.
* Replace the existing nameservers with the ones from Route 53.



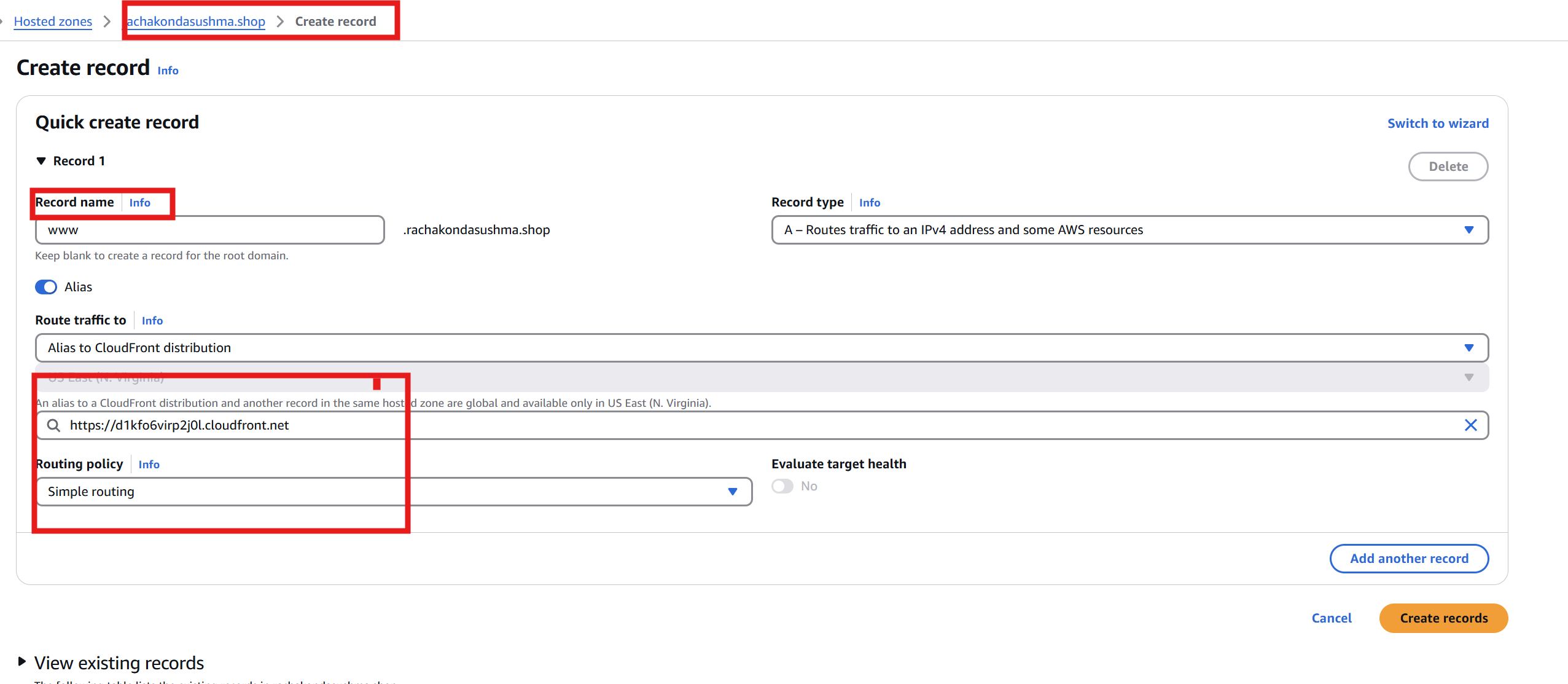
Step 3: Create a CloudFront Distribution (CDN)

* Go to the **CloudFront Console** → <https://console.aws.amazon.com/cloudfront/>
* Click **“Create Distribution”**
* Under **Origin**, set the S3 bucket, ALB, or other service as the origin.
* In the **Alternate Domain Name (CNAME)** field, add your custom domain (e.g., www.example.com)
* **Attach SSL/TLS Certificate**:
* Choose **“Custom SSL Certificate”**
* Request a certificate from **AWS Certificate Manager (ACM)** in **us-east-1**
* Click **“Create Distribution”**



Step 4: Create a Record Set in Route 53 to Map Domain to CloudFront

* Go back to **Route 53 → Hosted Zone → example.com**
  + - Click **"Create record"**
* Set:
  + **Record name**: e.g., www (for www.example.com)
  + **Record type**: A – IPv4 address
  + **Alias**: Yes
  + **Alias target**: Choose your CloudFront distribution from the list
* Click **"Create records"**



6) Update the index.html in s3 bucket and the updated file should be accessible by using domain name.

Step 1: Update index.html in S3 Bucket

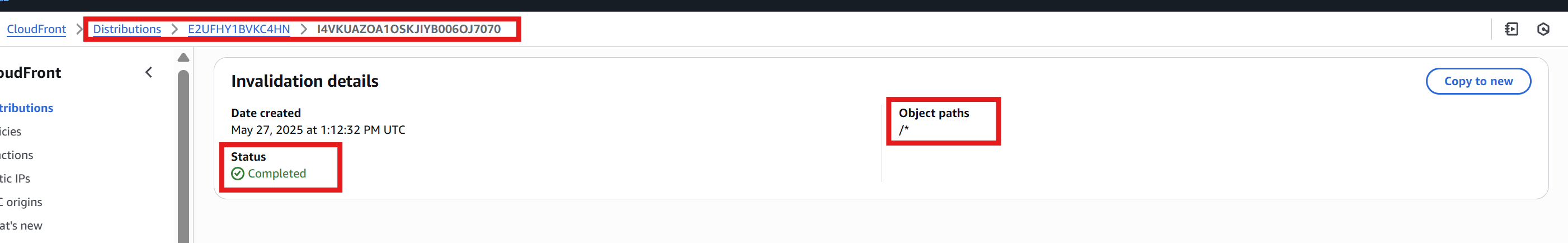
* + - Go to the **S3 Console**
    - Open your bucket
    - Navigate to index.html
    - Click **Upload** → select the updated index.html file
    - Choose **"Overwrite"** if prompted
    - Click **"Upload"**



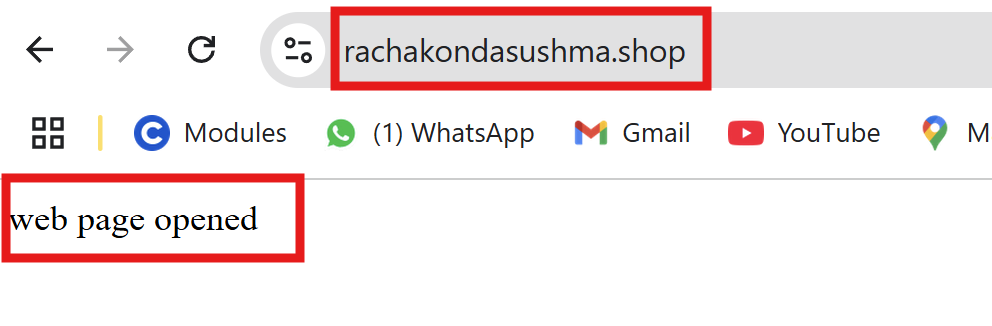
Step 2: Invalidate CloudFront Cache (for immediate changes)

CloudFront caches content, so to reflect changes immediately:

* + - Go to the **CloudFront Console**
    - Select your distribution
    - Go to **"Invalidations"** tab
    - Click **"Create Invalidation"**
    - Enter /\* (or /index.html if that's all you're updating)
    - Click **"Invalidate"**



Step 3: Test with Custom Domain



7) Share the Domain name in slack to test the connectivity.

Mydomain:

<https://rachakondsushma.shop/>

