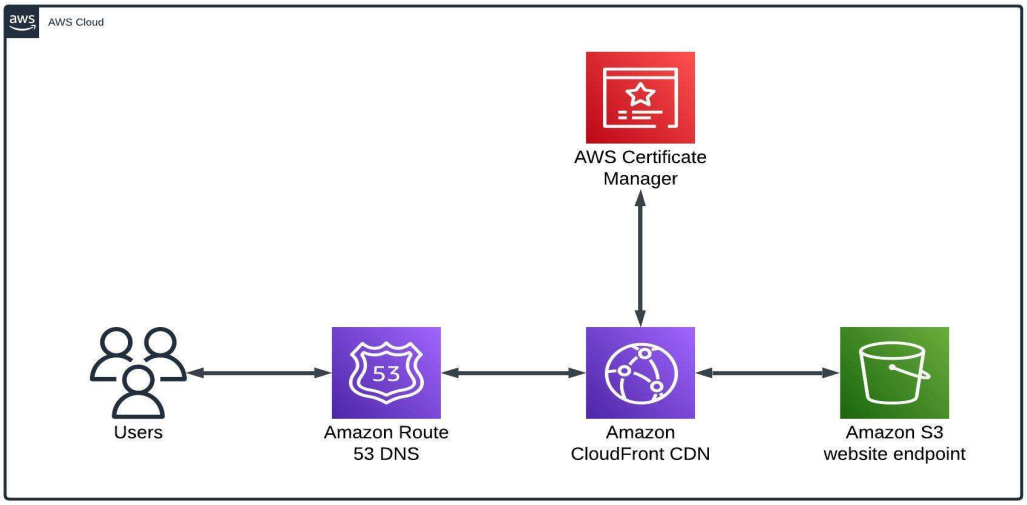
Static Website Hosting on Amazon S3 using CloudFront Distribution and Route 53.



Standard Amazon S3 Website Hosting Architecture

STEPS

A. Creation of an index.html file

B. Create an S3 bucket and set it up for static website hosting.

C. Create a record set in Route 53

D. AWS Certificate Manager SSL Public Certificates Request and Secure the connection via SSL.

E. Set up a CloudFront distribution and link it with a custom domain.

F. Configure cache invalidation of CloudFront.

G. Create Alias records in Route 53.

STEP 1: Creation of an index.html file.

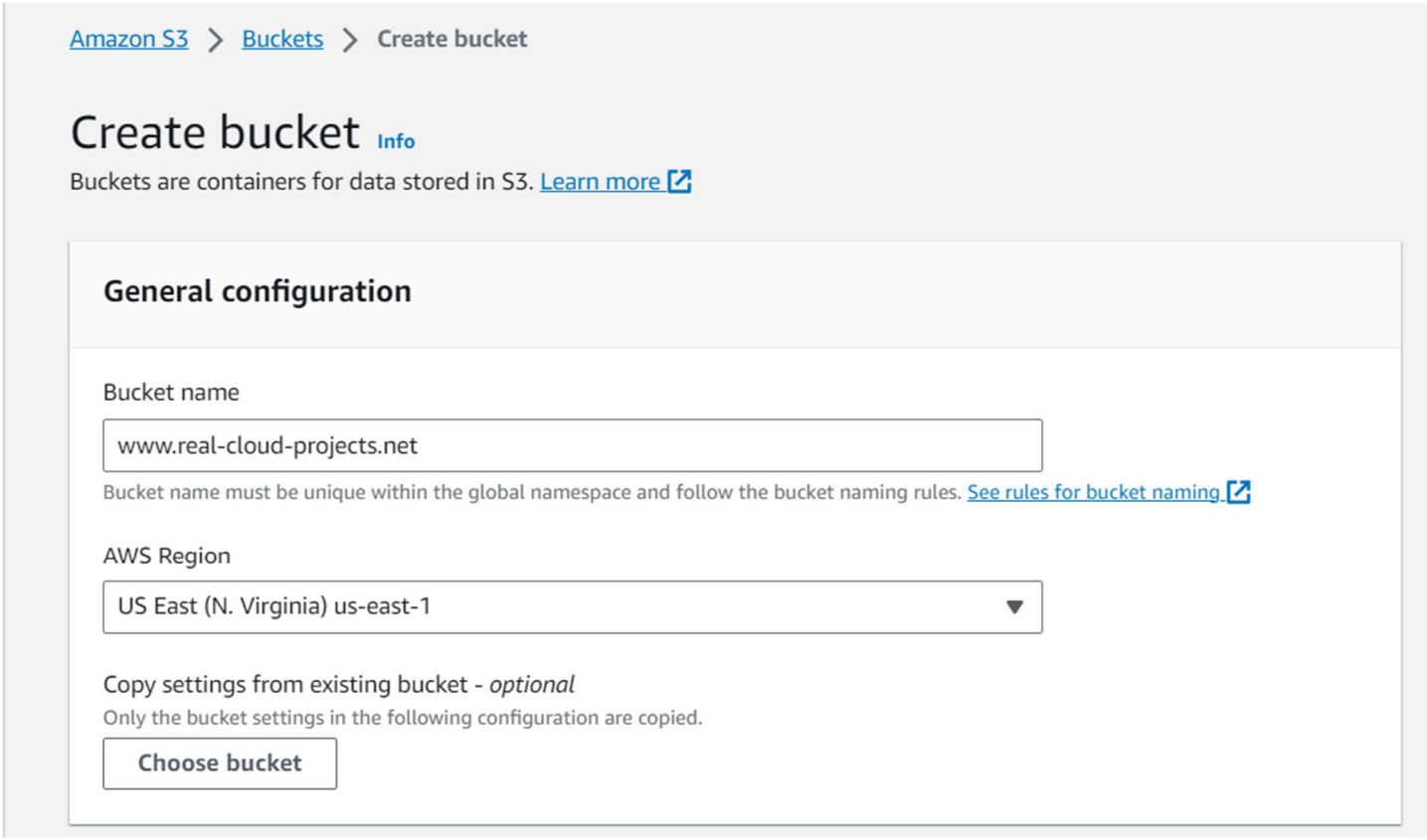
The **index.html** file serves as the default file that is displayed when a visitor accesses the root domain or subdirectory of your website, and it handles the URL route entry point for your website. The index.html file acts as a default page, manages errors, and supports the functionality of single-page applications in S3 static website hosting.

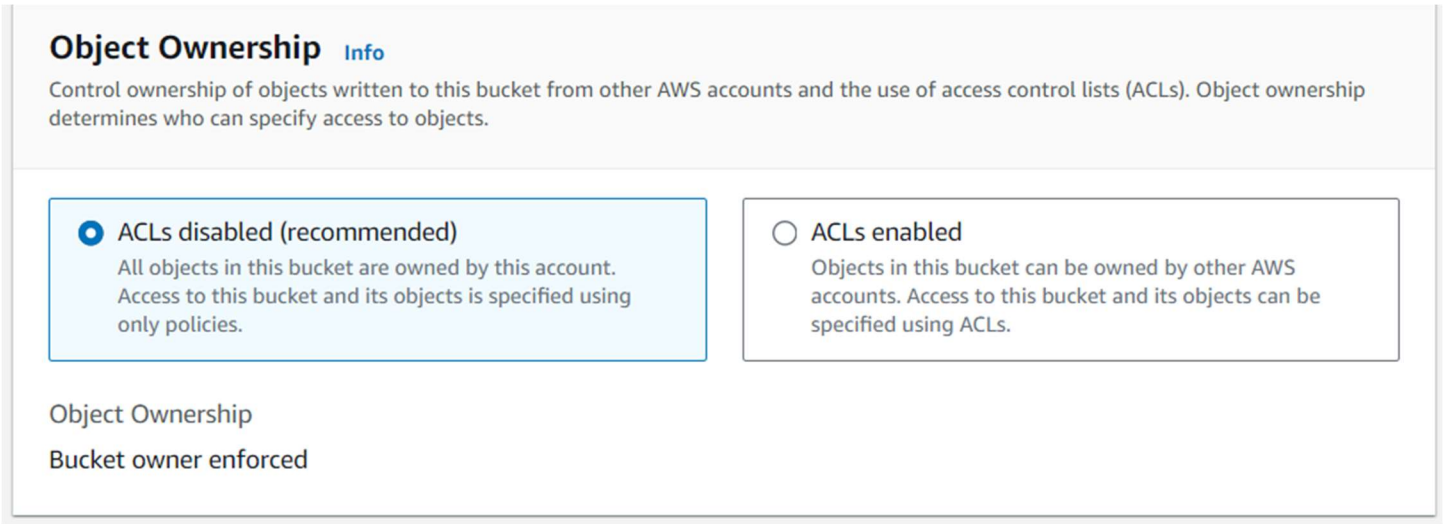
STEP 2:

Create an S3 bucket with public access

Bucket name MUST be same as your registered domain name,

for example, **www.real-cloud-projects.net** is my bucket name.





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

- [x] **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.

  - [x] **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.

  - [x] **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.

  - [x] **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.

  - [x] **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.

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**

    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.

- [x] I acknowledge that the current settings might result in this bucket and the

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

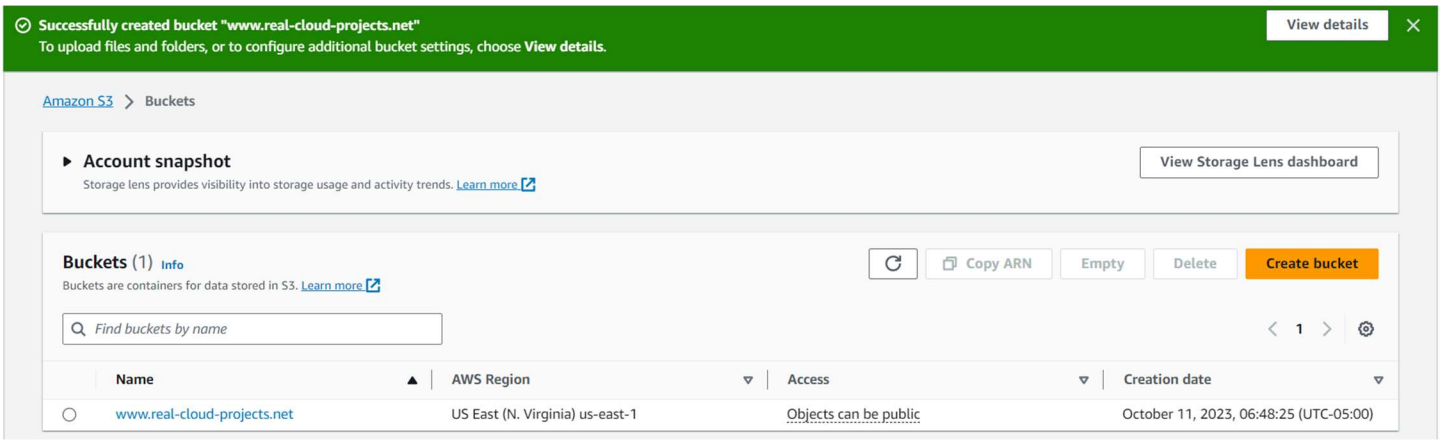
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. Learn more

- ( ) 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



My **S3 bucket** www.real-cloud-projects.net was successfully created.

STEP 3:

Let's configure Static Website Hosting for our S3 Bucket.

Click on the S3 bucket **www.real-cloud-projects.net**

Click on **Properties**

Scroll all way down to the bottom of the page to **Static website hosting**

Click on **Edit**

Under Static Website Hosting

Select **Enable**

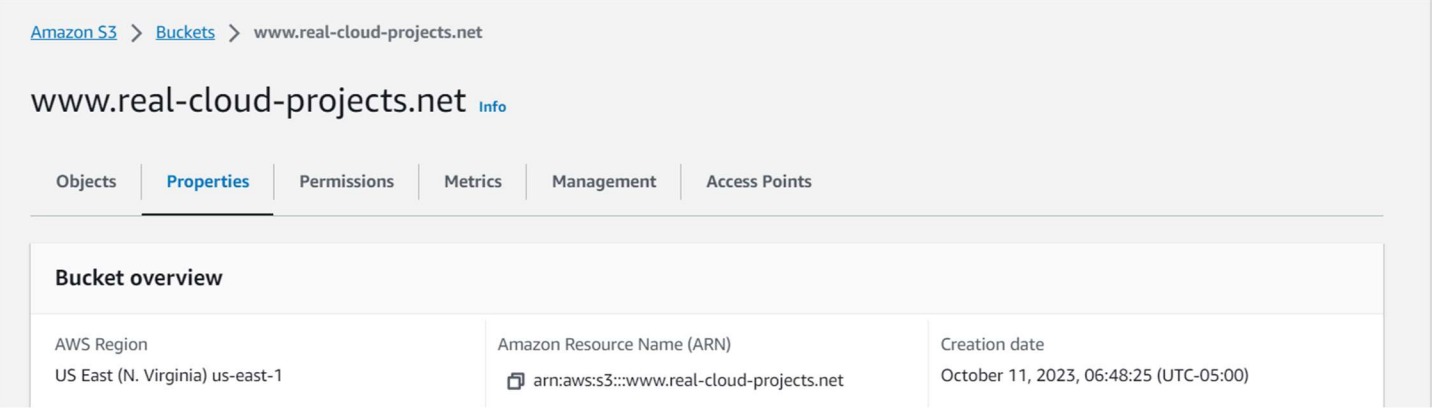
Under Hosting type

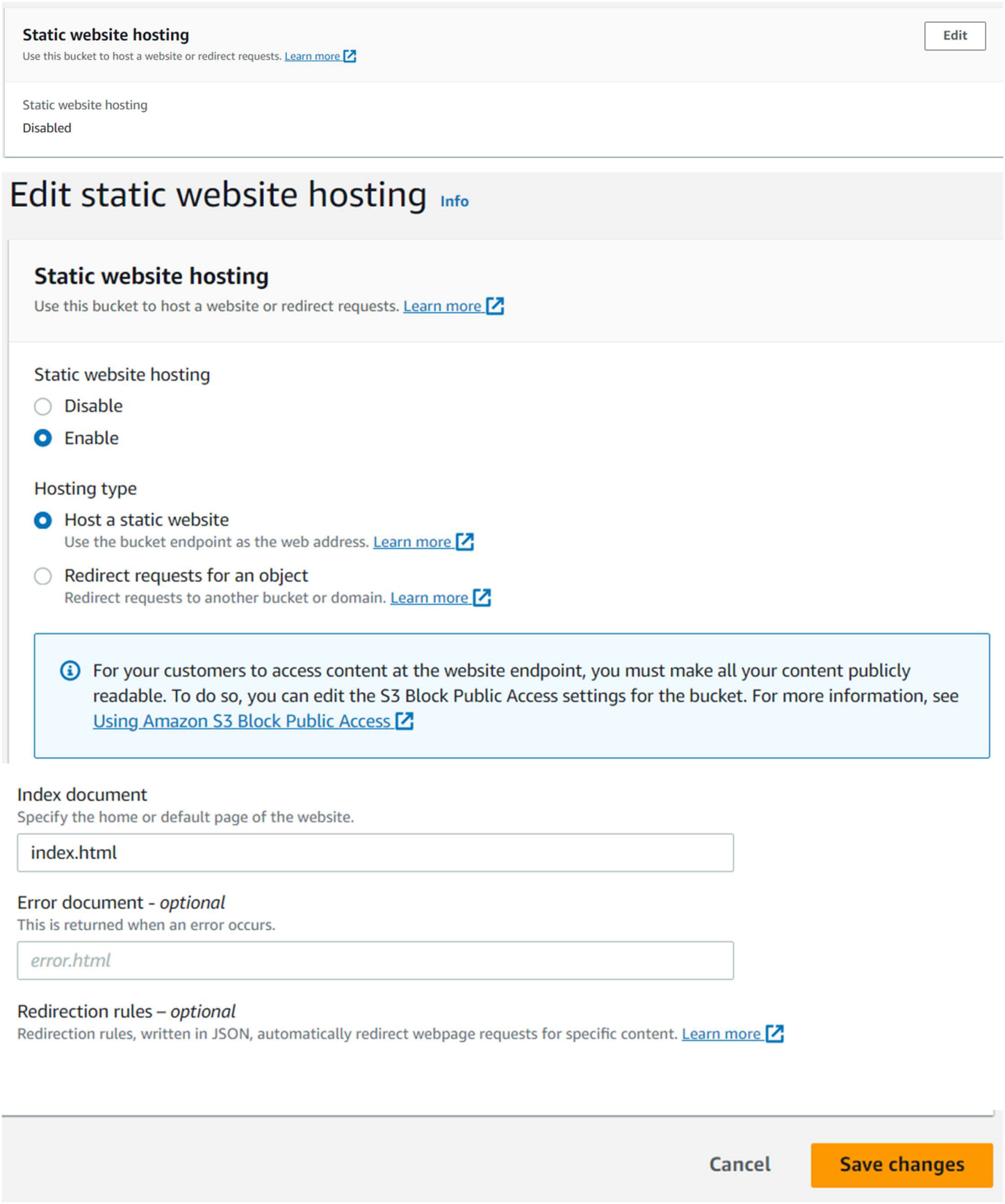
Select **Host a static a website**

Under Index document

Enter index.html

Error document **– optional**





Click on "Save changes"

Successfully edited static website hosting for my S3 bucket **www.real-cloud-projects.net**

STEP 4:

The next step will be to upload an **index.html** file to the S3 bucket.

I will also upload an image.png file and a **resume.pdf** file to the S3 bucket.

Make sure the pdf file is named correctly, for example "Solutions_Architect_Resume.pdf"

Click on the **S3 bucket**

Click on **Upload** tab

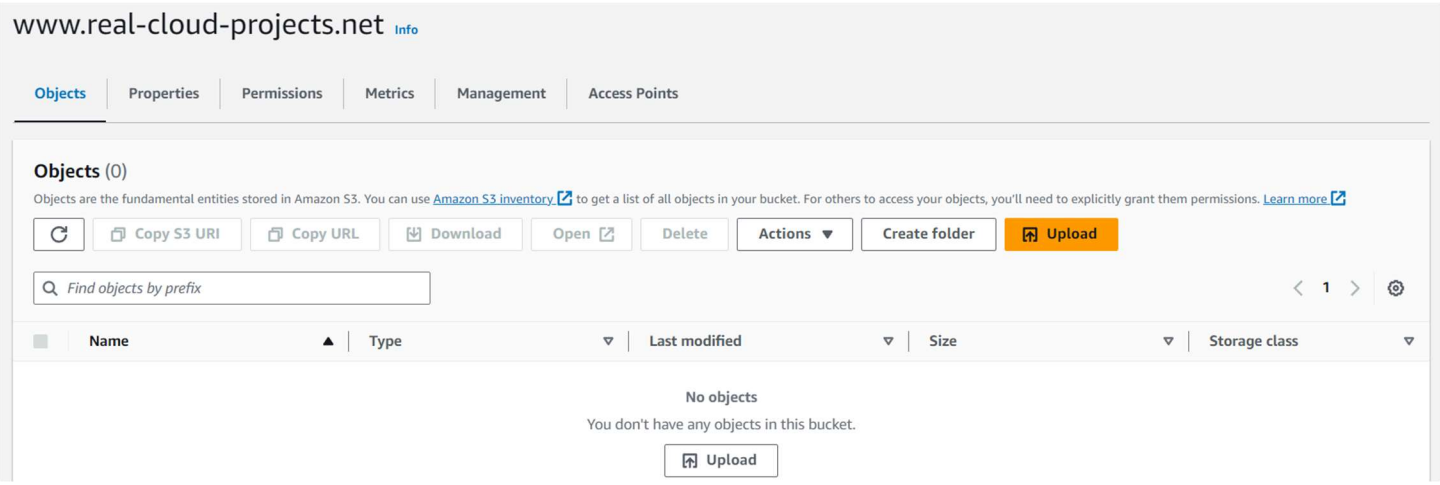
Click on **Add file**

Navigate to file **destination folder** to select the file

In my case, I will navigate desktop to select **index.html** file

Click on Upload tab, to upload the selected files unto S3 bucket

Repeat the same process for uploading images, files, folders etc.



Upload Info

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (3 Total, 192.6 KB)

Remove | Add files | Add folder

All files and folders in this table will be uploaded.

Find by name

1

Name	Folder	Type	Size
index.html	-	text/html	467.0 B
Solutions_Architect_...	-	application/pdf	189.9 KB
wave.png	-	image/png	2.3 KB

Destination

Destination

s3://www.real-cloud-projects.net

▶ **Destination details**

Bucket settings that impact new objects stored in the specified destination.

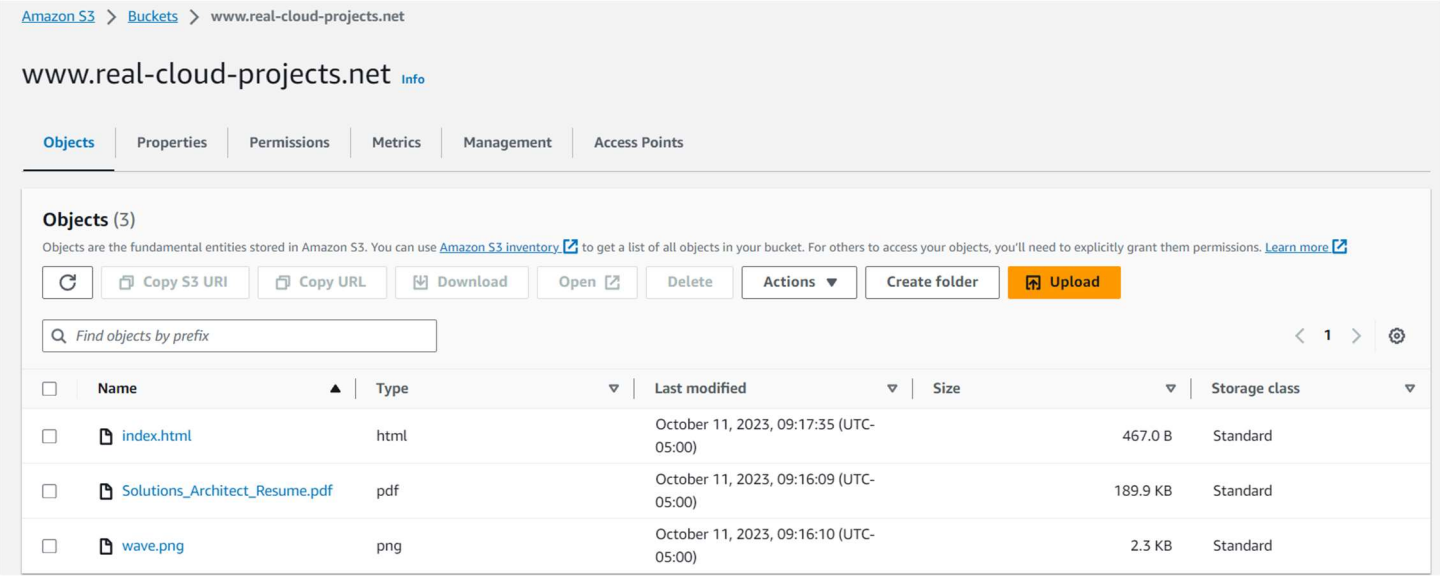
▶ **Permissions**

Grant public access and access to other AWS accounts.

▶ **Properties**

Specify storage class, encryption settings, tags, and more.

Cancel | Upload



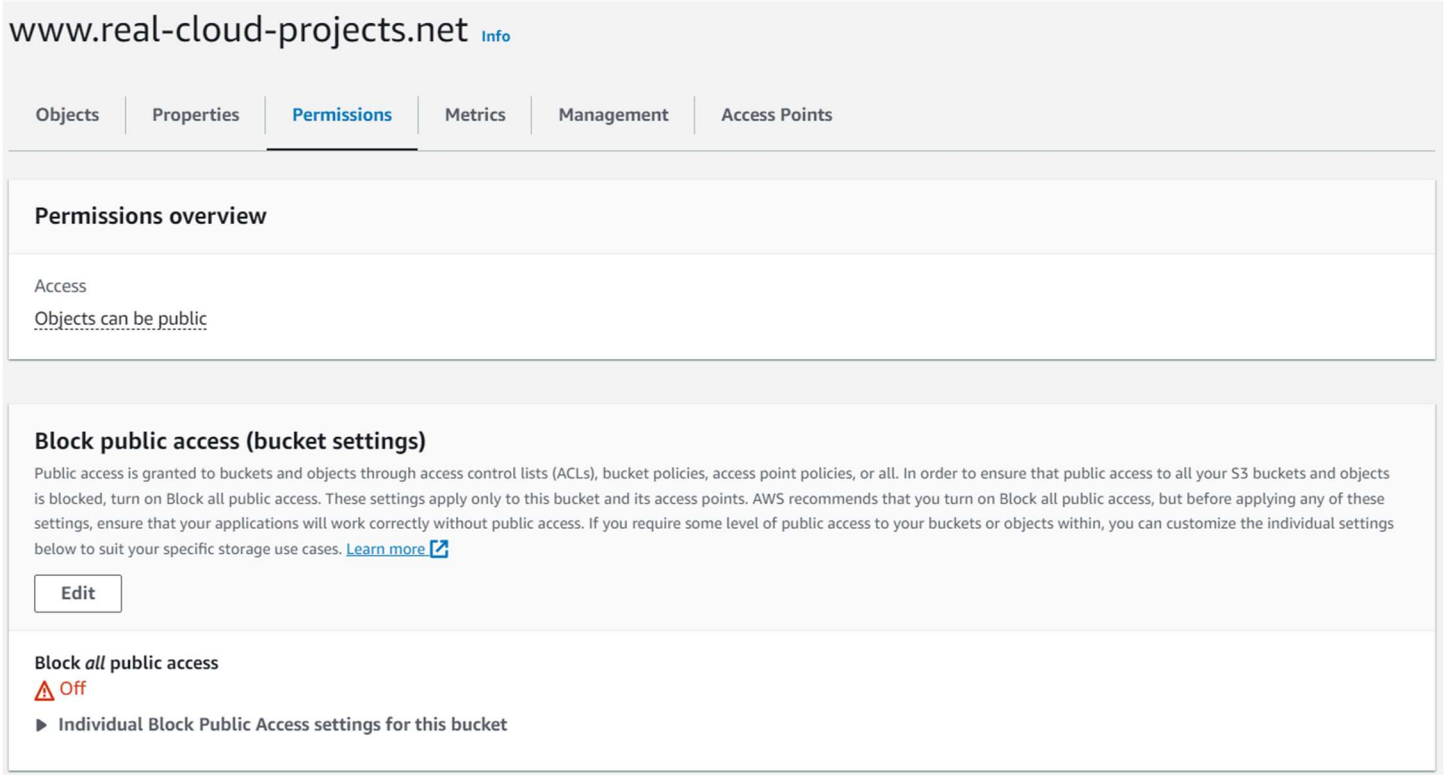
The index.html document, wave image.png and Solutions_Architect_Resume.pdf were successfully uploaded to the S3 bucket.

STEP 5:

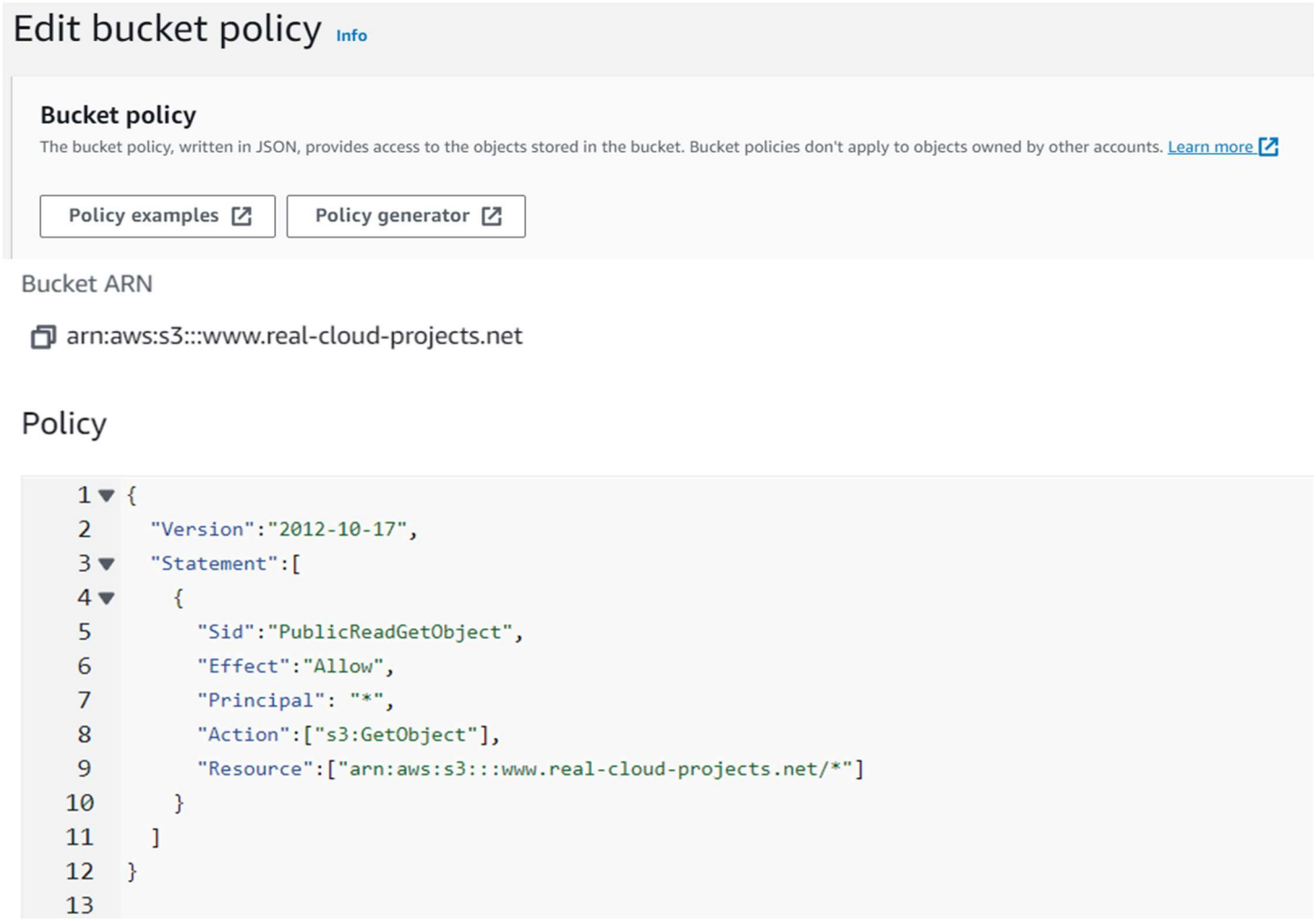
In this stage, we will generate a bucket policy which will allow the bucket to be publicly accessed.

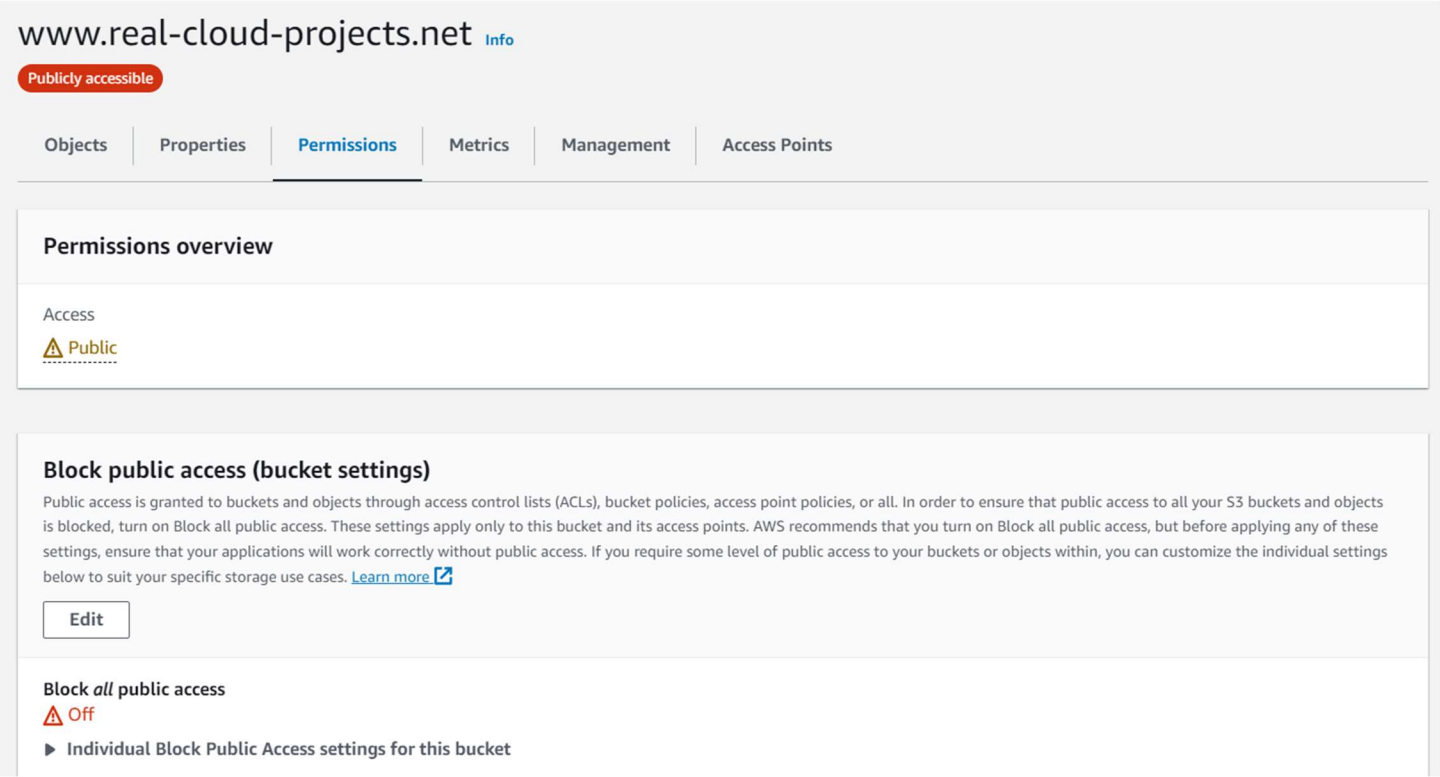
Copy bucket ARN in the format of arn:aws:s3:::<bucket name> and paste it in the "Resource":["arn:aws:s3:::YOUR_BUCKET_NAME/*"] of the policy, that will allow public access to all objects stored in the bucket.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "PublicReadGetObject",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:GetObject",
   "Resource": "arn:aws:s3:::www.real-cloud-projects.net/*"
  }
 ]
}
```



Edit bucket policy and add the S3 bucket policy to grant public access.

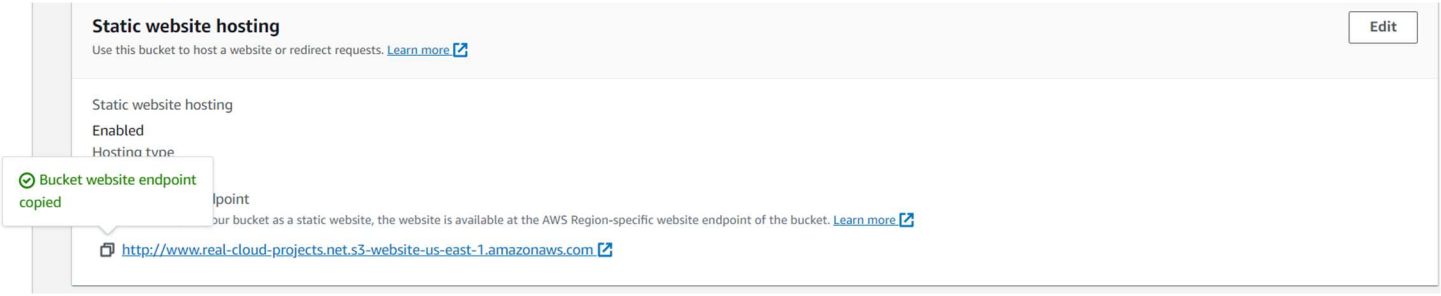




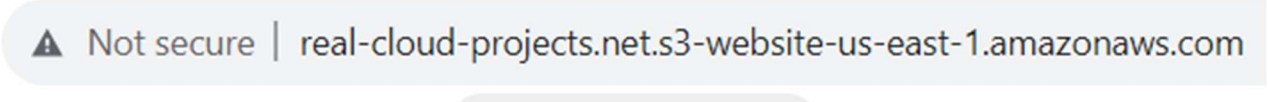
From the above screenshot, we can see that we have **public access**, which means that our S3 bucket can be publicly accessed by anonymous users.

Now, let's click on the **Properties** tab
Let's scroll down to the bottom of the page
Under Static Website Hosting, **copy** the "**bucket website endpoint**" URL.

http://www.real-cloud-projects.net.s3-website-us-east-1.amazonaws.com



Now, I will paste the bucket website endpoint on a new browser tab, now the http unsecure website will be displayed.



Hello!

My name is Edwin Esene

Welcome to My Static Web Page Hosted on Amazon S3

My resume is hosted here for your access and review!

click here to download resume

Thank you for visiting.

The Static website hosting configuration was successful as shown in the above screenshot. The website is public and can be accessed by any anonymous user.

STEP 6:
IMPORTANT:
Recall that our static website can only be reached via **www.real-cloud-projects.net** at this moment.

Now let's configure a second S3 bucket **real-cloud-projects.net** to redirect requests traffic from our non "www" **real-cloud-projects.net** URL to our www.real-cloud-projects.net URL.

Create bucket Info

Buckets are containers for data stored in S3. Learn more

General configuration

Bucket name

real-cloud-projects.net

Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming

AWS Region

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

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Choose bucket

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

- [x] **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.

  - [x] **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.

  - [x] **Block public access to buckets and objects granted through *any* access control lists (ACLs)**

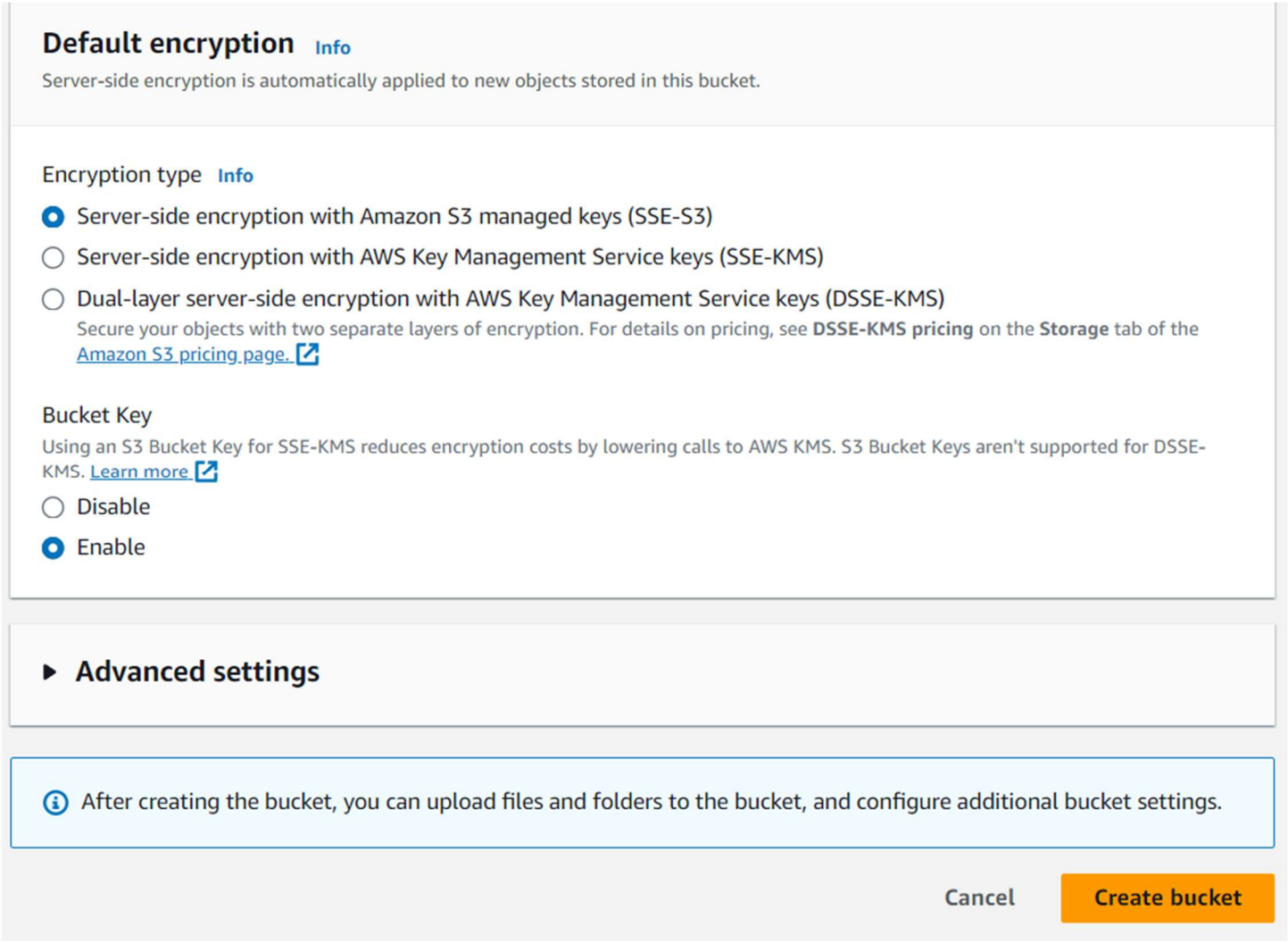
    S3 will ignore all ACLs that grant public access to buckets and objects.

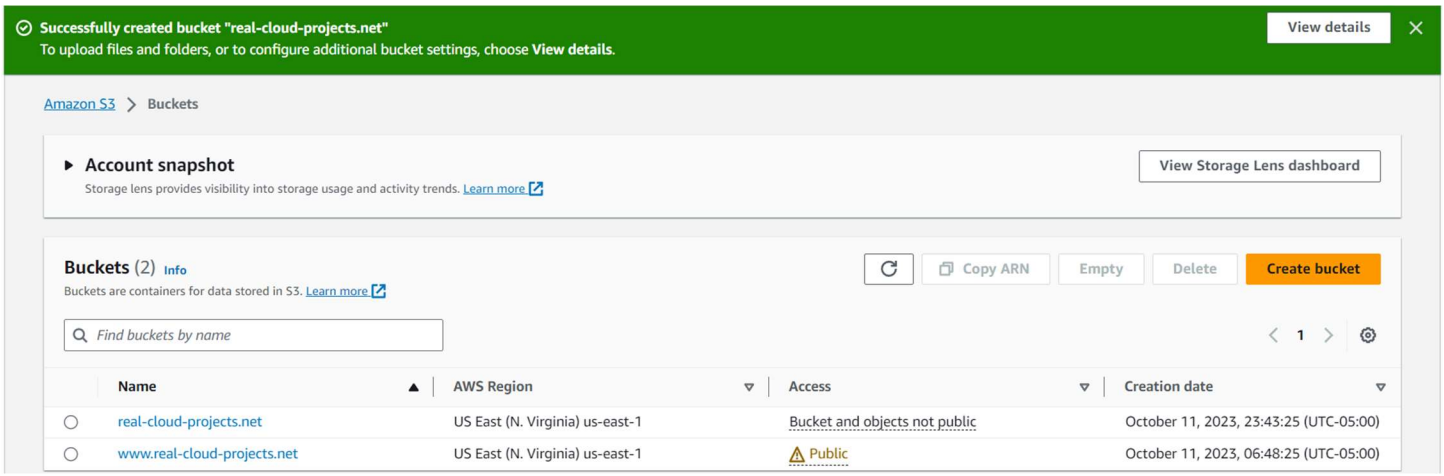
  - [x] **Block public access to buckets and objects granted through *new* public bucket or access point policies**

    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.

  - [x] **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.





S3 bucket is successfully created as shown in the above screenshot.

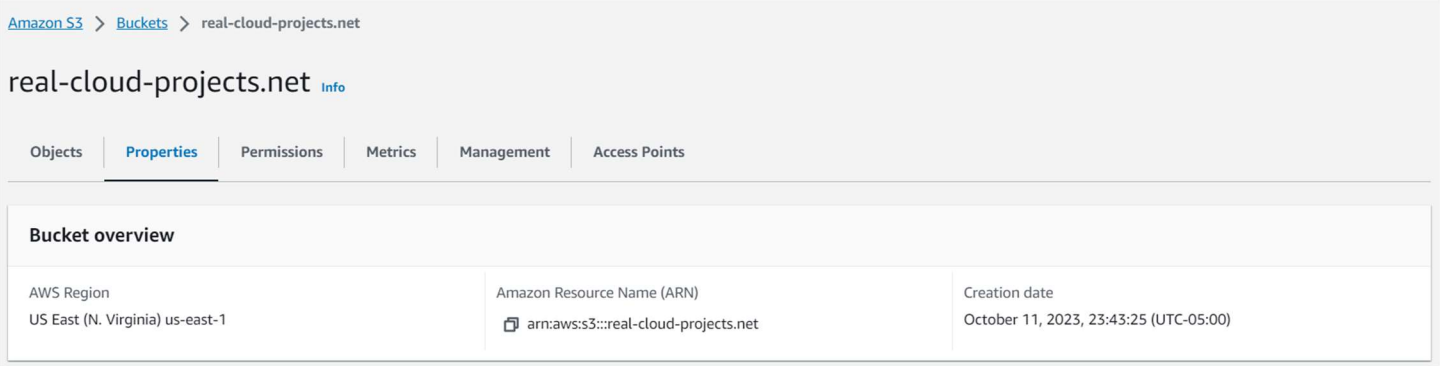
To configure traffic redirection from real-cloud-projects.net to www.real-cloud-projects.net
To configure static website:

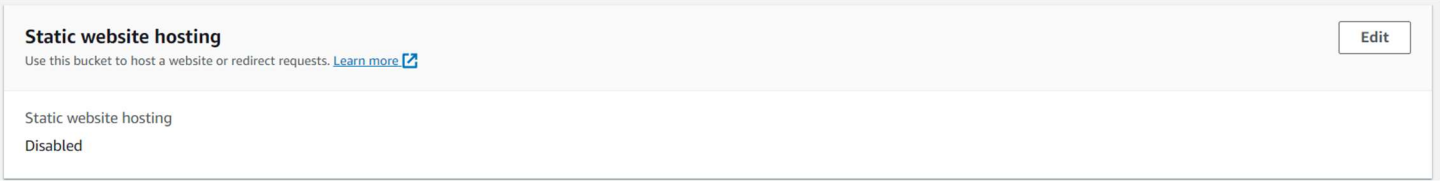
Click on the bucket "**real-cloud-projects.net**" to open bucket
Click on the "**Properties**" tab
Scroll down to the bottom of page

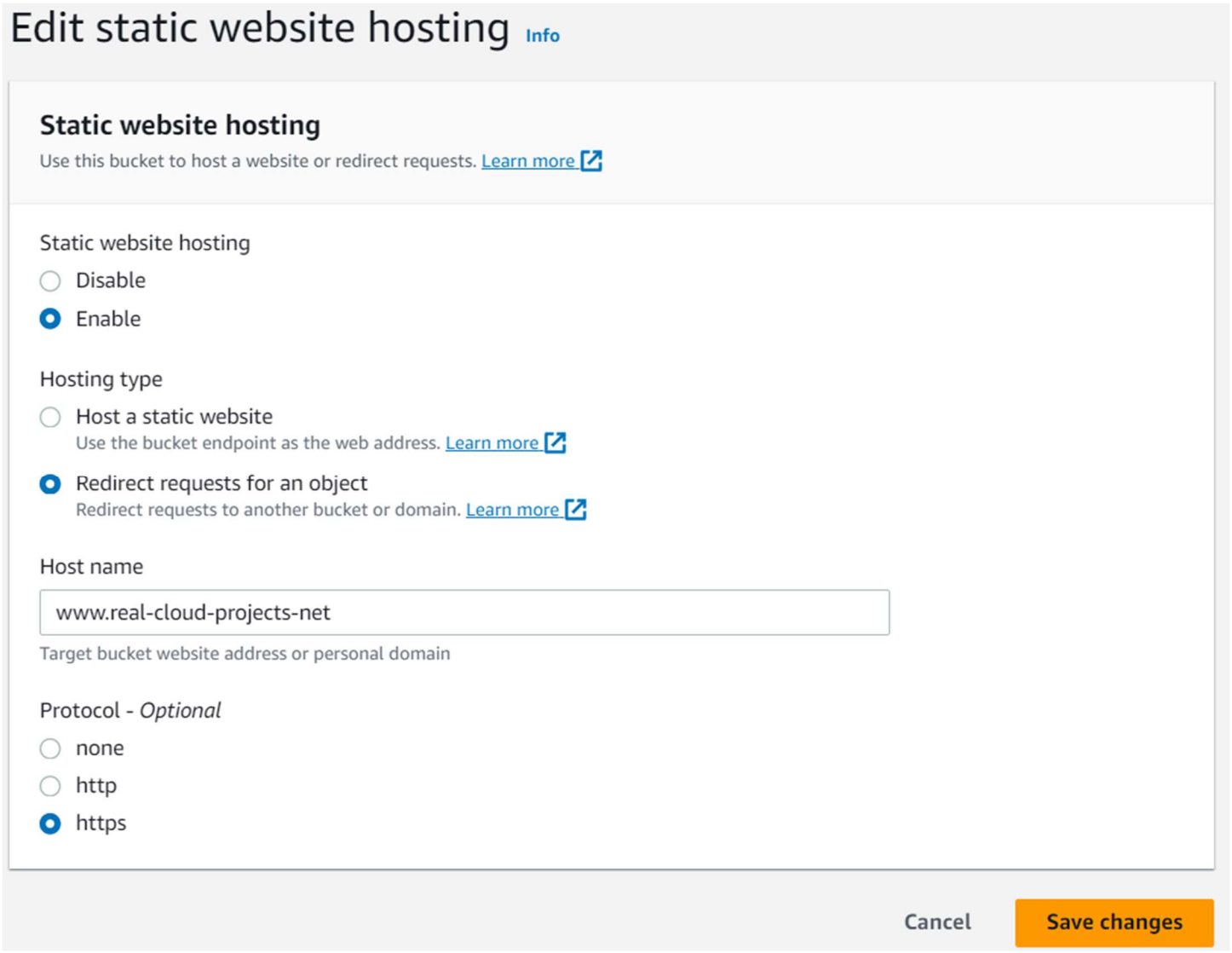
Click on "**Static website hosting**"
Click on "**Edit**"
Select "**Enable Static website hosting**"
Select "**Redirect request for an object**"

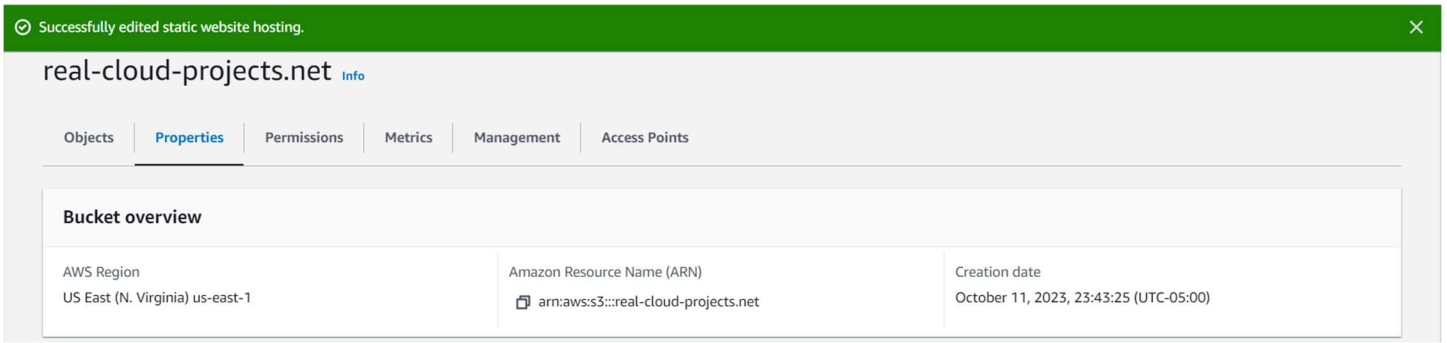
This will redirect requests from our "**non** www" real-cloud-projets.net URL to our www.real-cloud-projects.net URL.

Select the "**https**" protocol





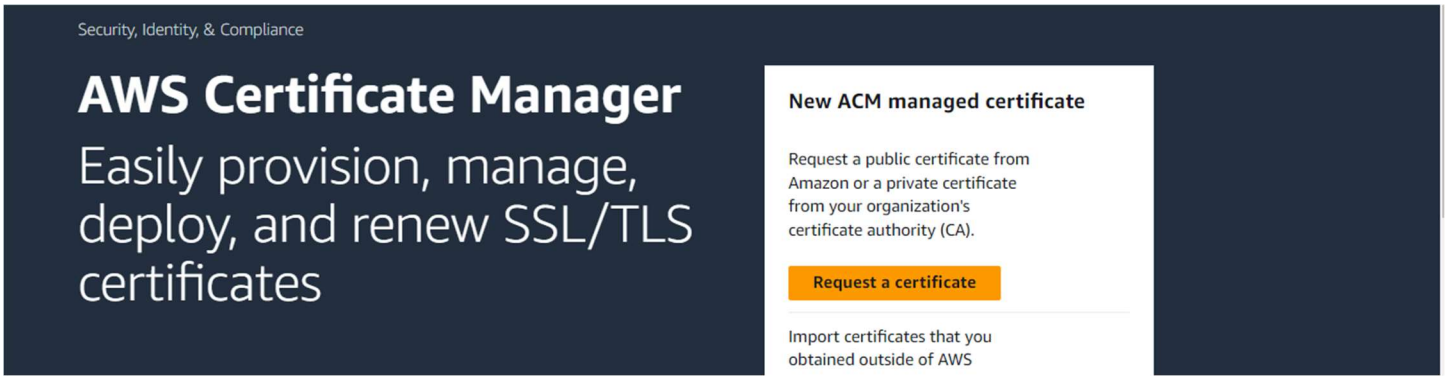




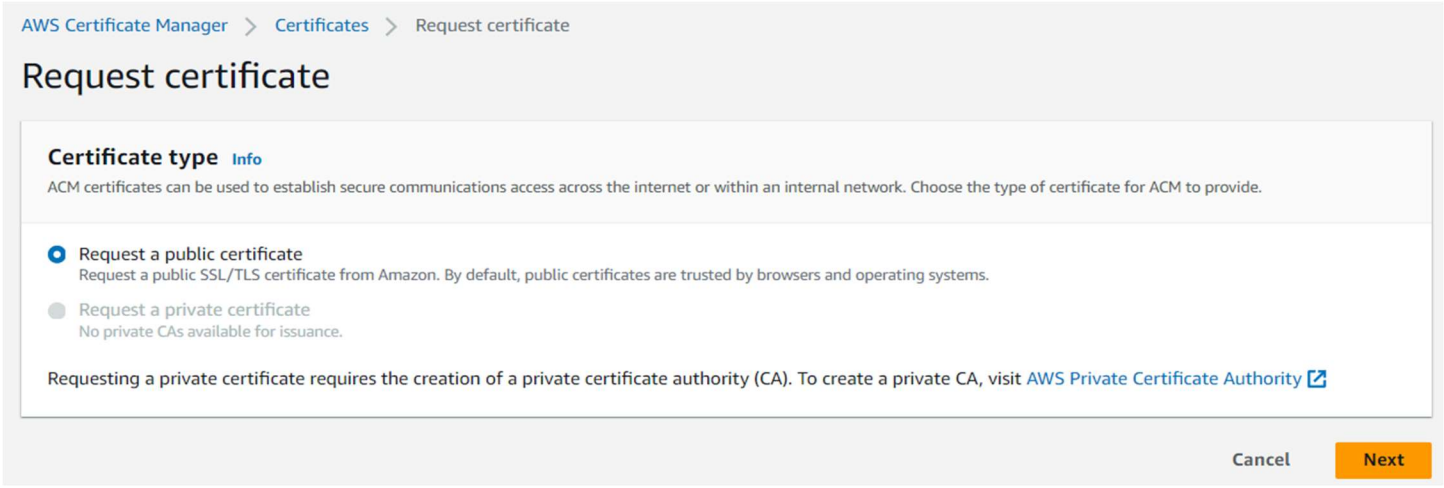
The configuration of my second S3 bucket "**real-cloud-projects.net**" for static website hosting was successful as shown above screenshots.

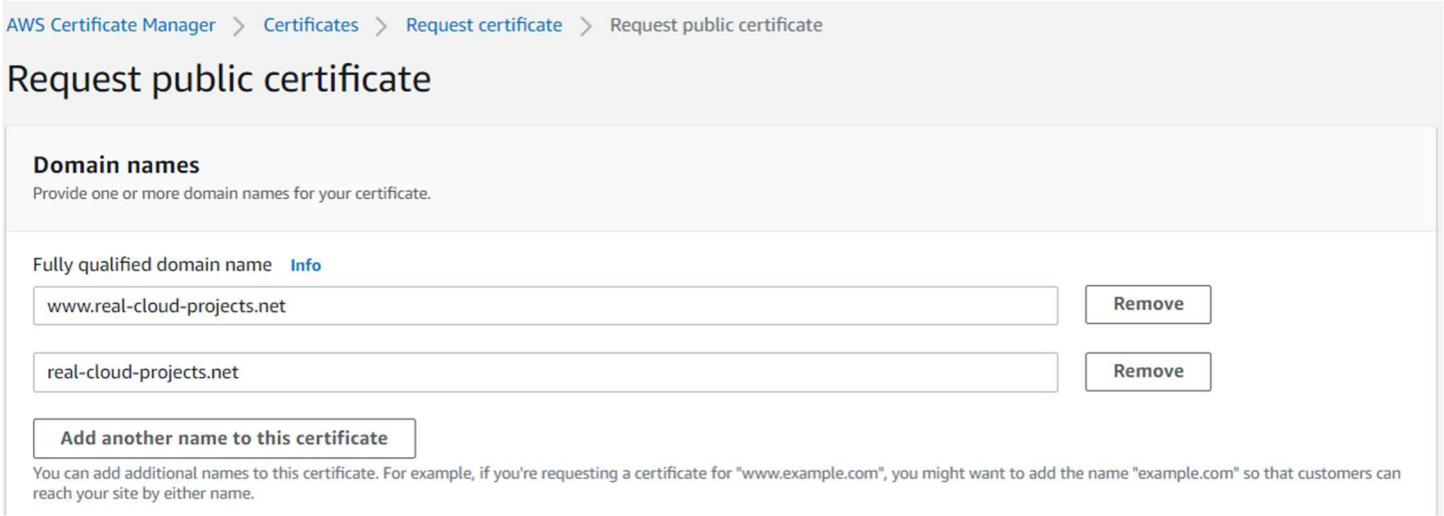
STEP 7:
AWS Certificate Manager (ACM) to request SSL Public Certificates

It is important to select the AWS North Virginia region (US-East-1) when requesting a public certificate because CloudFront only recognizes this region as it's ACM certificates issuer.



Click Request a public certificate to Request a certificate.
Fill in the domain name **www.real-cloud-projects.net** in my case for this project
I will also enter another domain name as **real-cloud-projects.net** for this project.





Validation method Info

Select a method for validating domain ownership.

- (●) DNS validation - recommended
  Choose this option if you are authorized to modify the DNS configuration for the domains in your certificate request.
- ( ) Email validation
  Choose this option if you do not have permission or cannot obtain permission to modify the DNS configuration for the domains in your certificate request.

Key algorithm Info

Select an encryption algorithm. Some algorithms may not be supported by all AWS services.

- (●) RSA 2048
  RSA is the most widely used key type.
- ( ) ECDSA P 256
  Equivalent in cryptographic strength to RSA 3072.
- ( ) ECDSA P 384
  Equivalent in cryptographic strength to RSA 7680.

Tags Info

To help you manage your certificates, you can optionally assign your own metadata to each resource in the form of tags.

No tags associated with this resource.

Add tag

You can add 50 more tag(s).

Cancel | Previous | Request

AWS Certificate Manager > Certificates > f4354bb1-e1b3-4d24-8f33-82cac082b01c

f4354bb1-e1b3-4d24-8f33-82cac082b01c

Delete

Certificate status

Identifier
f4354bb1-e1b3-4d24-8f33-82cac082b01c

Status
Pending validation Info

ARN
arn:aws:acm:us-east-1:113740152199:certificate/f4354bb1-e1b3-4d24-8f33-82cac082b01c

Type
Amazon Issued

Domains (2)

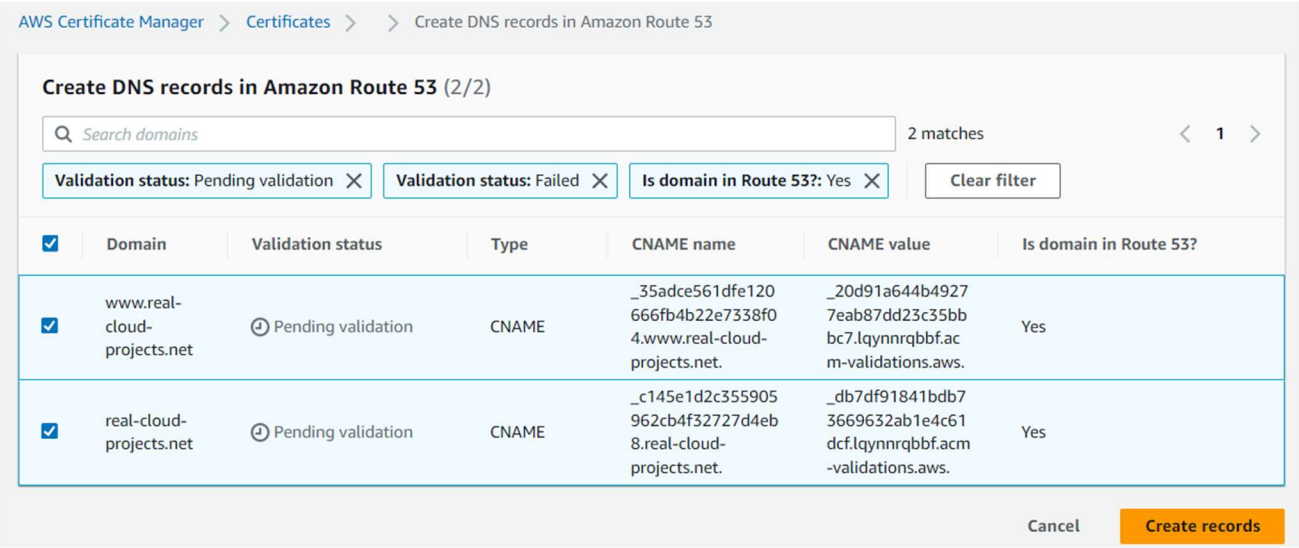
Create records in Route 53 | Export to CSV

1

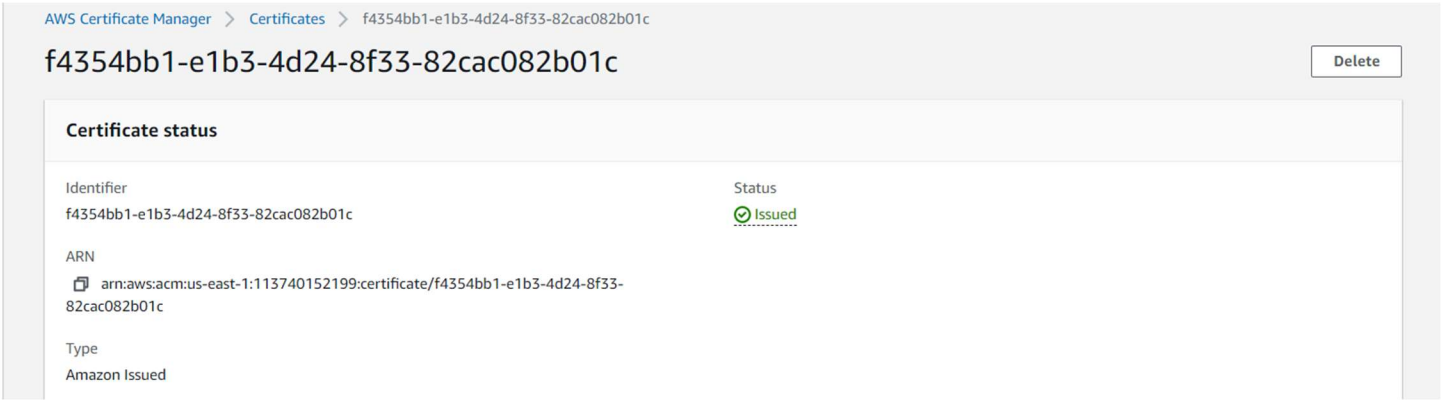
Domain	Status	Renewal status	Type	CNAME name	CNAME value
www.real-cloud-projects.net	Pending validation	-	CNAME	_35adce561dfe120666fb4b22e7338f04.www.real-cloud-projects.net.	_20d91a644b49277eab87dd23c35bbbc7.lqynnrqbbf.acm-validations.aws.
real-cloud-projects.net	Pending validation	-	CNAME	_c145e1d2c355905962cb4f32727d4eb8.real-cloud-projects.net.	_db7df91841bdb73669632ab1e4c61dcf.lqynnrqbbf.acm-validations.aws.

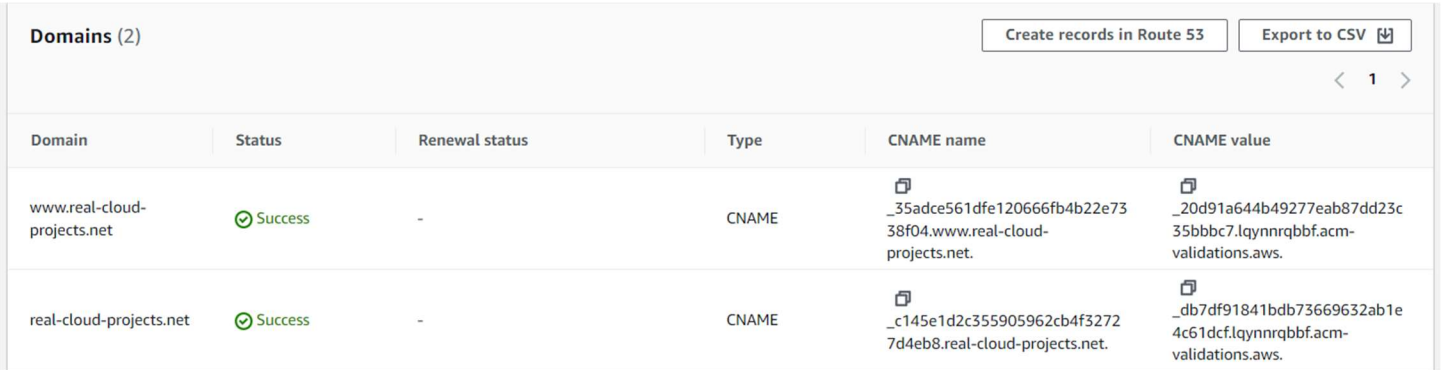
Choose a method of validation, **DNS validation** is the recommended option.
However, the certificates are still pending validation.
DNS validation entails the creation of CNAME records in Route 53 or any other DNS provider.

I already have Route 53 configured,
Click on **Create records in Route 53**



Click on **Create records**

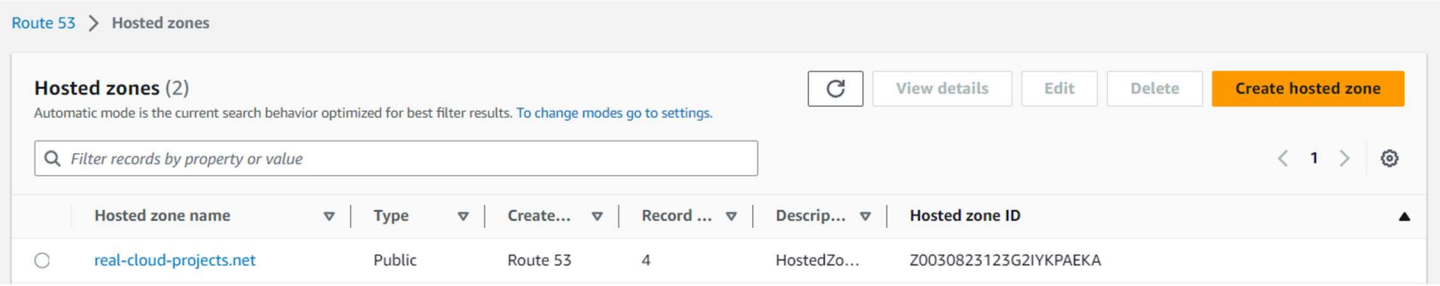




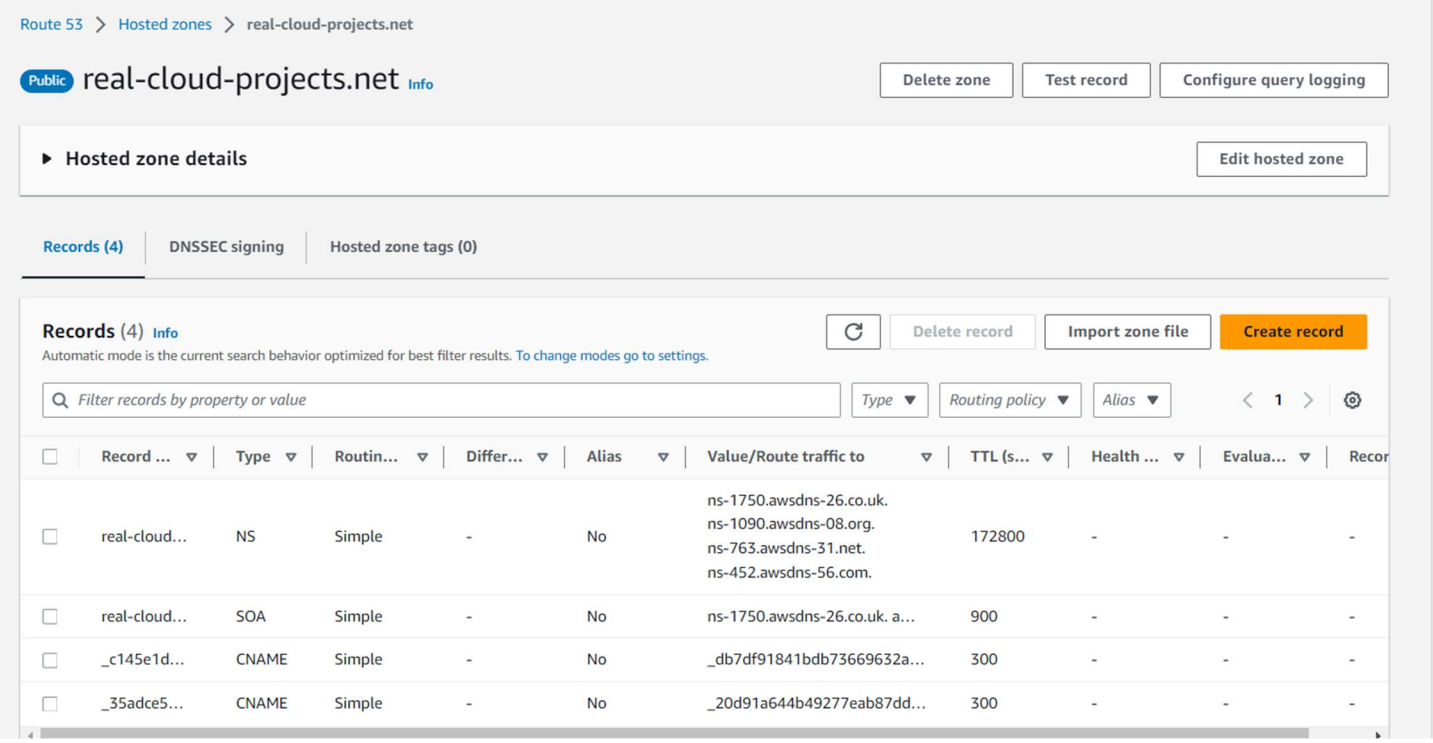
The DNS validation of certificate by the creation of CNAME records in Route 53 was successful.

STEP 8:
Navigate to Route 53, verify that CNAME records were successfully created.

Navigate to **Route 53**
Click on **Hosted zones**
Click on **real-cloud-projects.net**



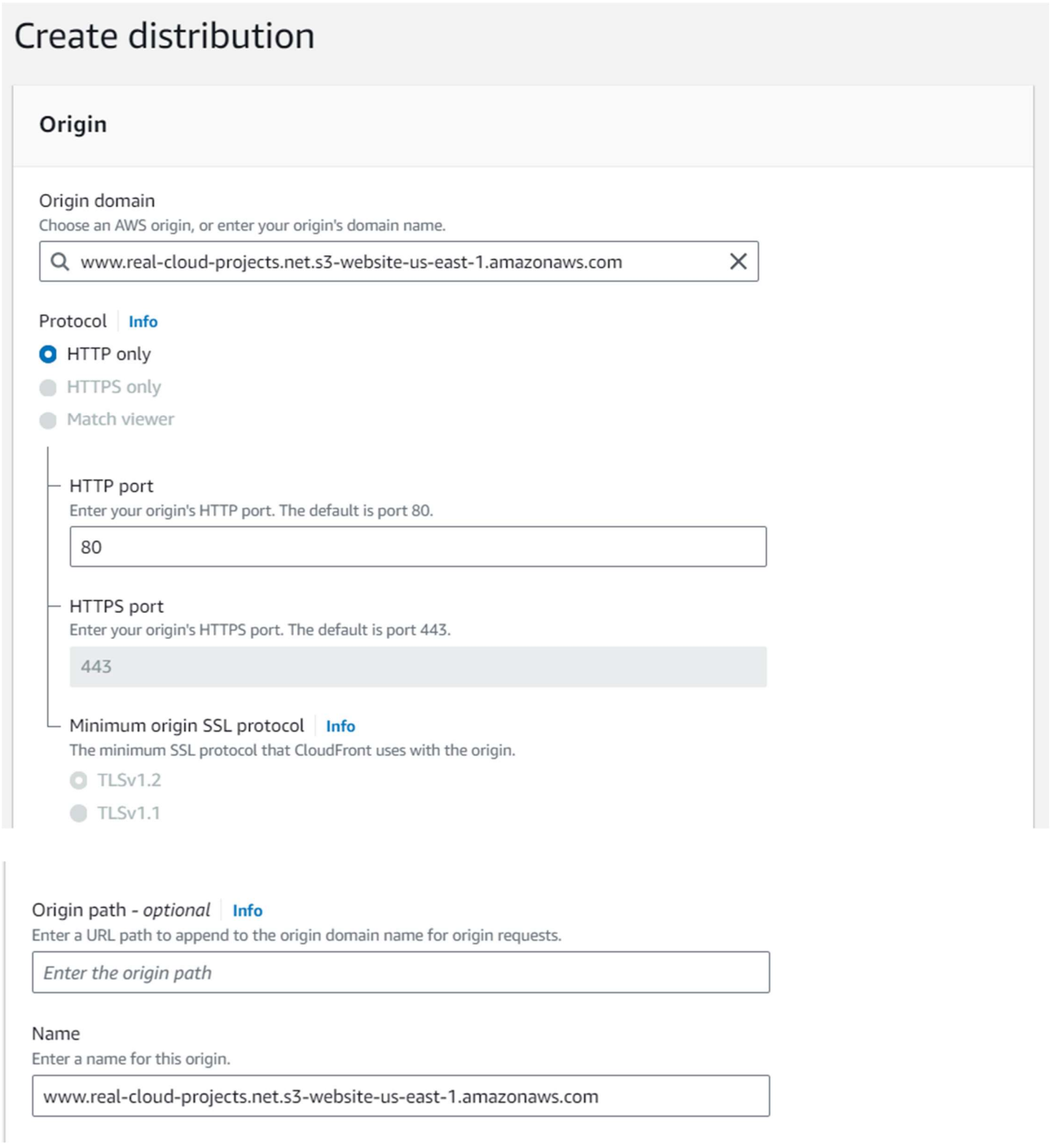
Looking at my hosted zone and refreshing it, I can see that two (2) CNAME Records have been recently created.



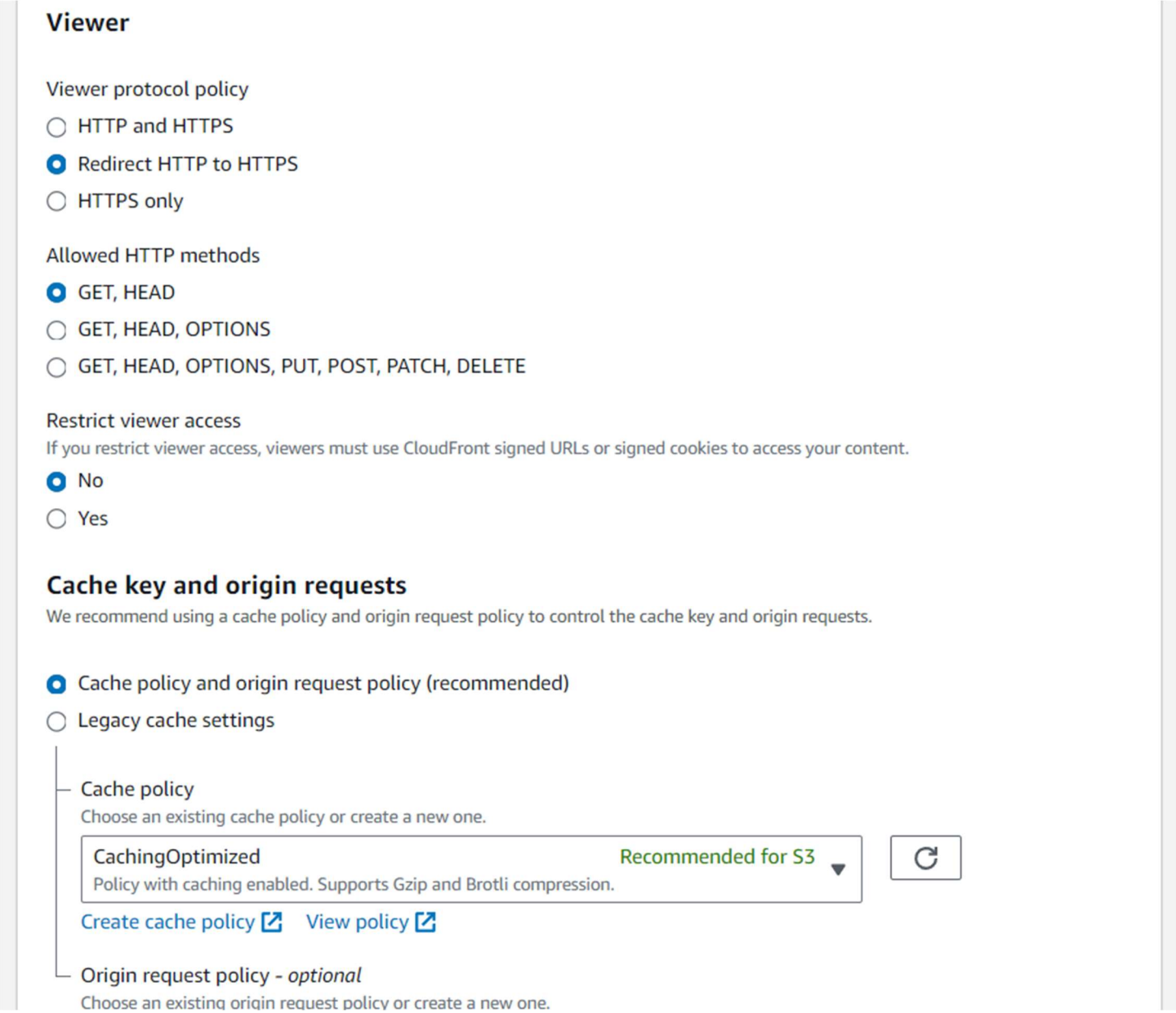
STEP 9:
Create a CloudFront distribution and link it with my custom domain.

In this section, we will configure a CloudFront distribution with an S3 bucket so that when an end-user tries to access my distribution, they will be redirected to my S3 contents, cached in edge locations via CloudFront.

For the Origin Domain Name, I will have to navigate back to my S3 management console, once in the S3 management console, I will click on my bucket name, then I will click on properties and scroll down to Static website hosting, then I will copy the bucket website Endpoint URL "**www.real-cloud-projects.net.s3-website-us-east-1.amazonaws.com**" in this case.
Now, I will go back to the CloudFront setup page and paste the URL inside the "**Origin Domain Name**" field.

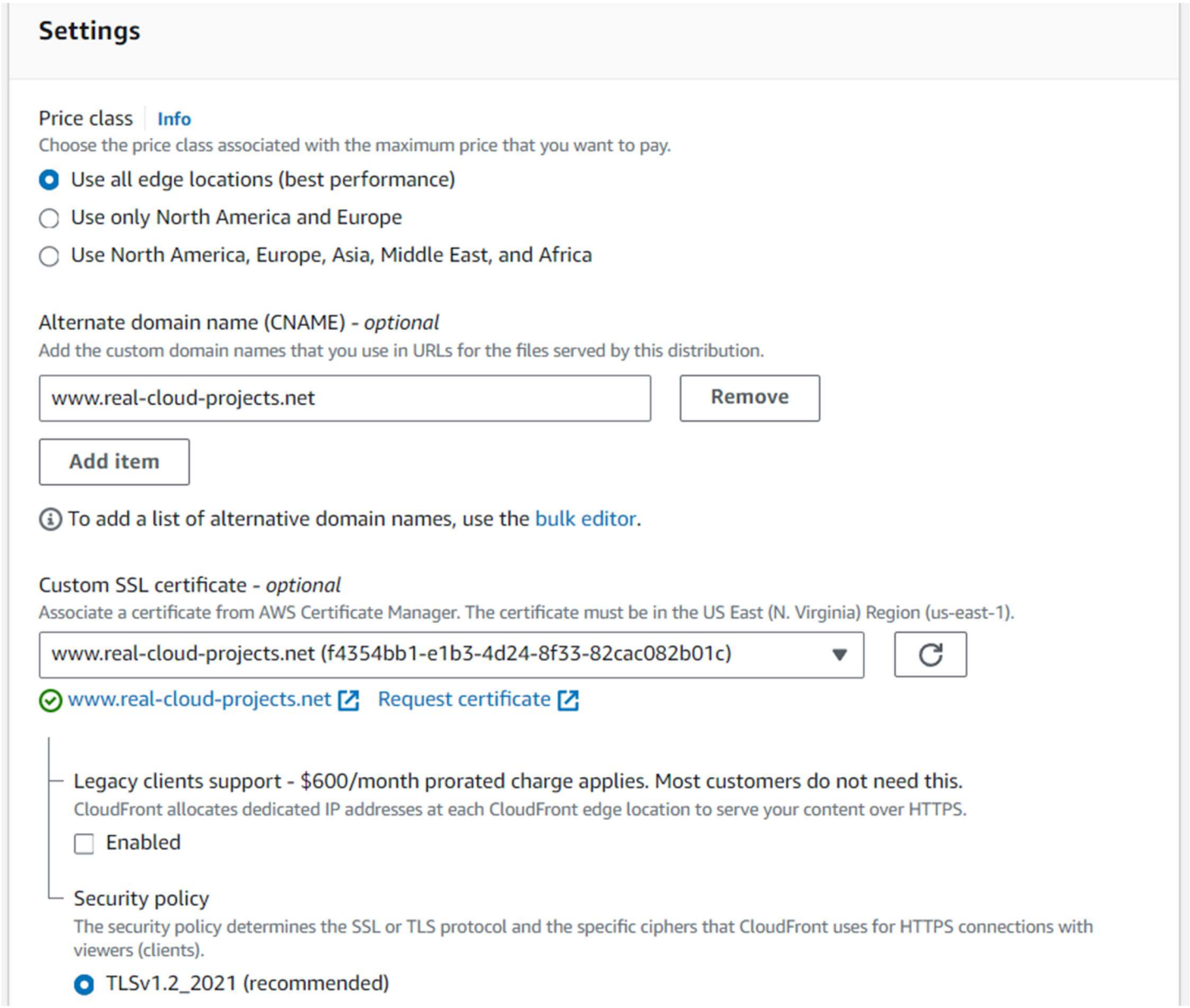


Under **Viewer** Protocol Policy, select **Redirect HTTP to HTTPS**

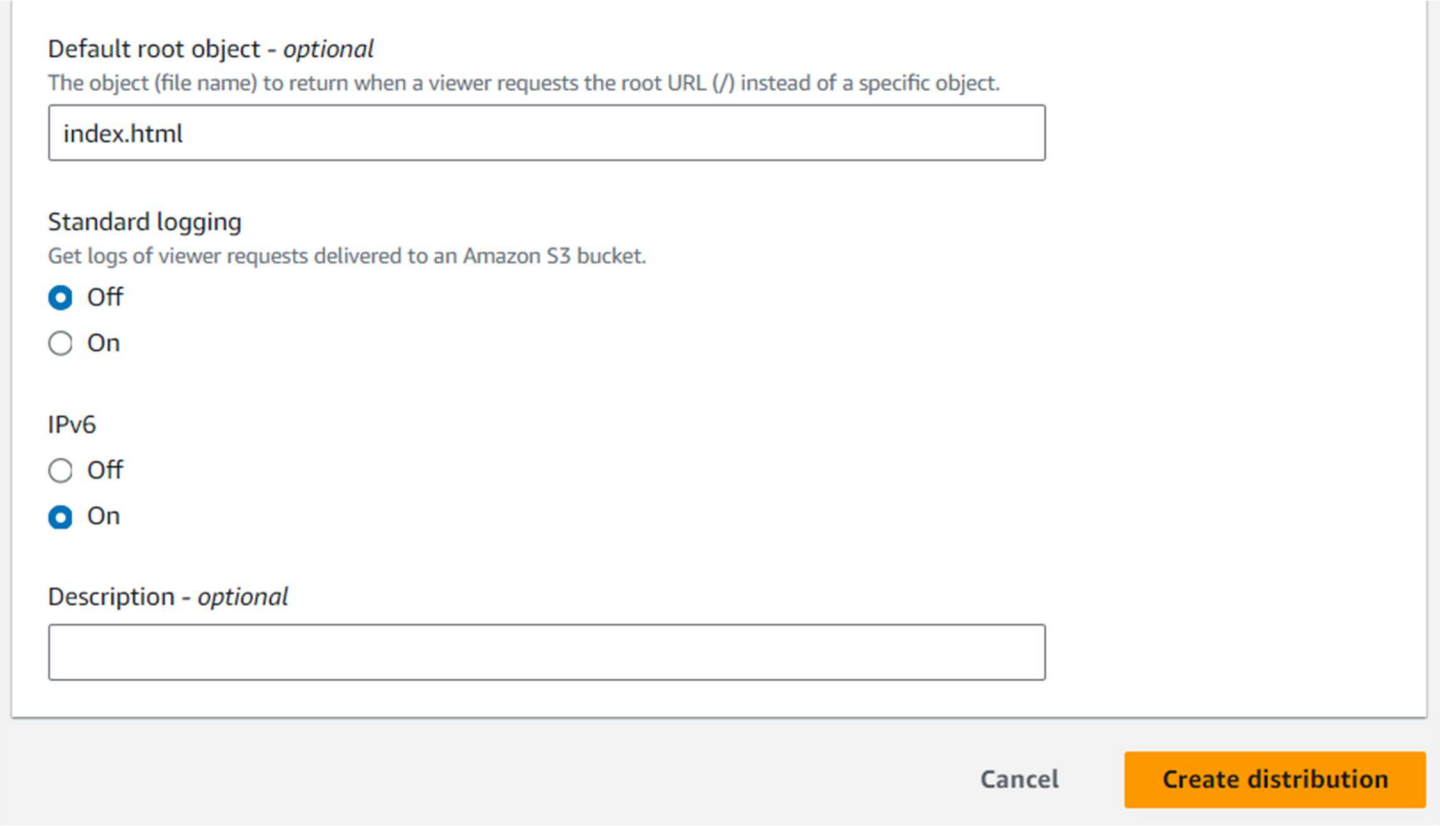


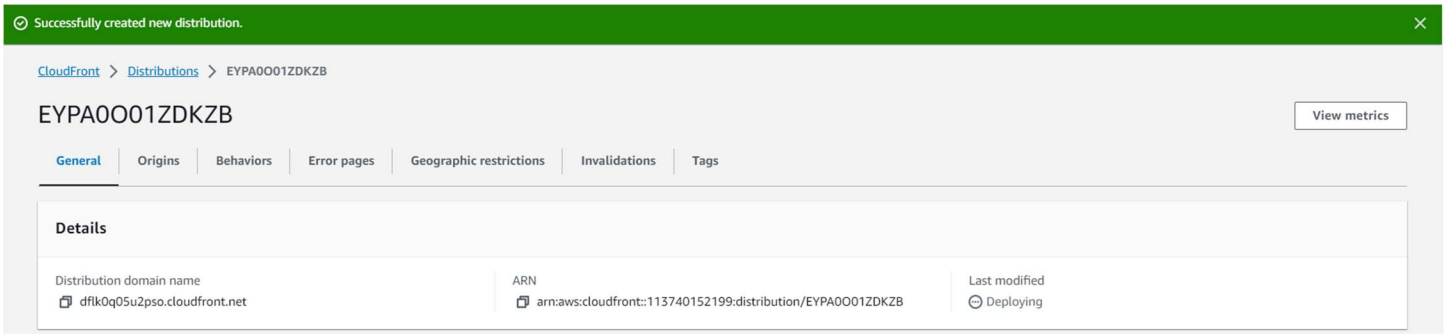
Next step will be to scroll down to Settings -> Alternate domain names (CNAMEs) and enter my domain name **www.real-cloud-projects.net** in the appropriate field.

We will select "**Custom SSL Certificate**" for CloudFront SSL encryption for desired field.
Scroll down to create the distribution.



Under **Default root object**, enter **index.html**
This means that whenever a user accesses our domain name, index.html will open as well.
Scroll down leaving all other fields as default
Click on **Create distribution**





The CloudFront distribution was successfully created.

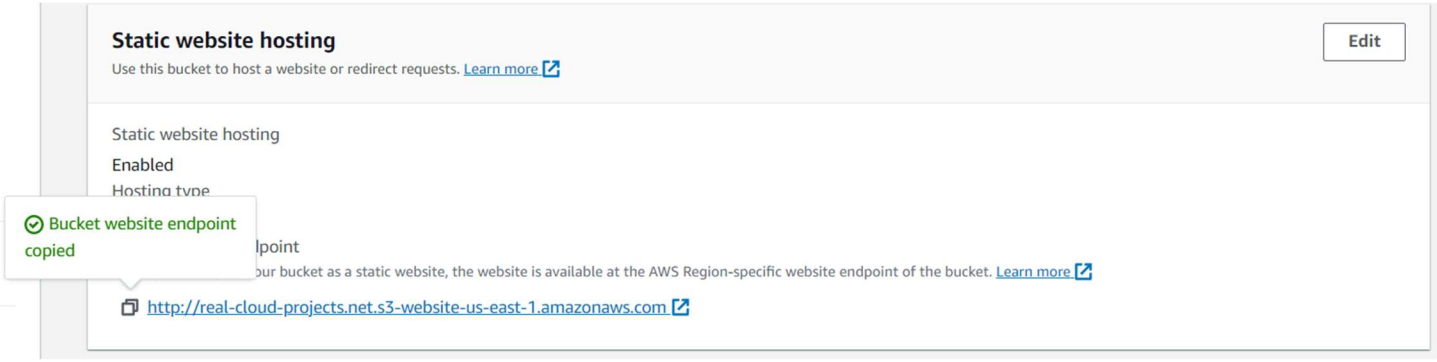
STEP 10:
Repeat the above same process plus second certificate to create a second CloudFront distribution with "real-cloud-projects.net"

real-cloud-projects.net Info

Objects | Properties | Permissions | Metrics | Management | Access Points

Bucket overview

AWS Region	Amazon Resource Name (ARN)	Creation date
US East (N. Virginia) us-east-1	arn:aws:s3:::real-cloud-projects.net	October 11, 2023, 23:43:25 (UTC-05:00)



Create distribution

Origin

Origin domain
Choose an AWS origin, or enter your origin's domain name.

real-cloud-projects.net.s3-website-us-east-1.amazonaws.com

Protocol | Info

- (•) HTTP only
- ( ) HTTPS only
- ( ) Match viewer

HTTP port
Enter your origin's HTTP port. The default is port 80.

80

HTTPS port
Enter your origin's HTTPS port. The default is port 443.

443

Minimum origin SSL protocol | Info
The minimum SSL protocol that CloudFront uses with the origin.

- ( ) TLSv1.2

Origin path - *optional* | Info
Enter a URL path to append to the origin domain name for origin requests.

Enter the origin path

Name
Enter a name for this origin.

real-cloud-projects.net.s3-website-us-east-1.amazonaws.com

Viewer

Viewer protocol policy

- ○ HTTP and HTTPS
- ◉ Redirect HTTP to HTTPS
- ○ HTTPS only

Allowed HTTP methods

- ◉ GET, HEAD
- ○ GET, HEAD, OPTIONS
- ○ GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE

Restrict viewer access

If you restrict viewer access, viewers must use CloudFront signed URLs or signed cookies to access your content.

- ◉ No
- ○ Yes

Cache key and origin requests

We recommend using a cache policy and origin request policy to control the cache key and origin requests.

- ◉ Cache policy and origin request policy (recommended)
- ○ Legacy cache settings

Cache policy

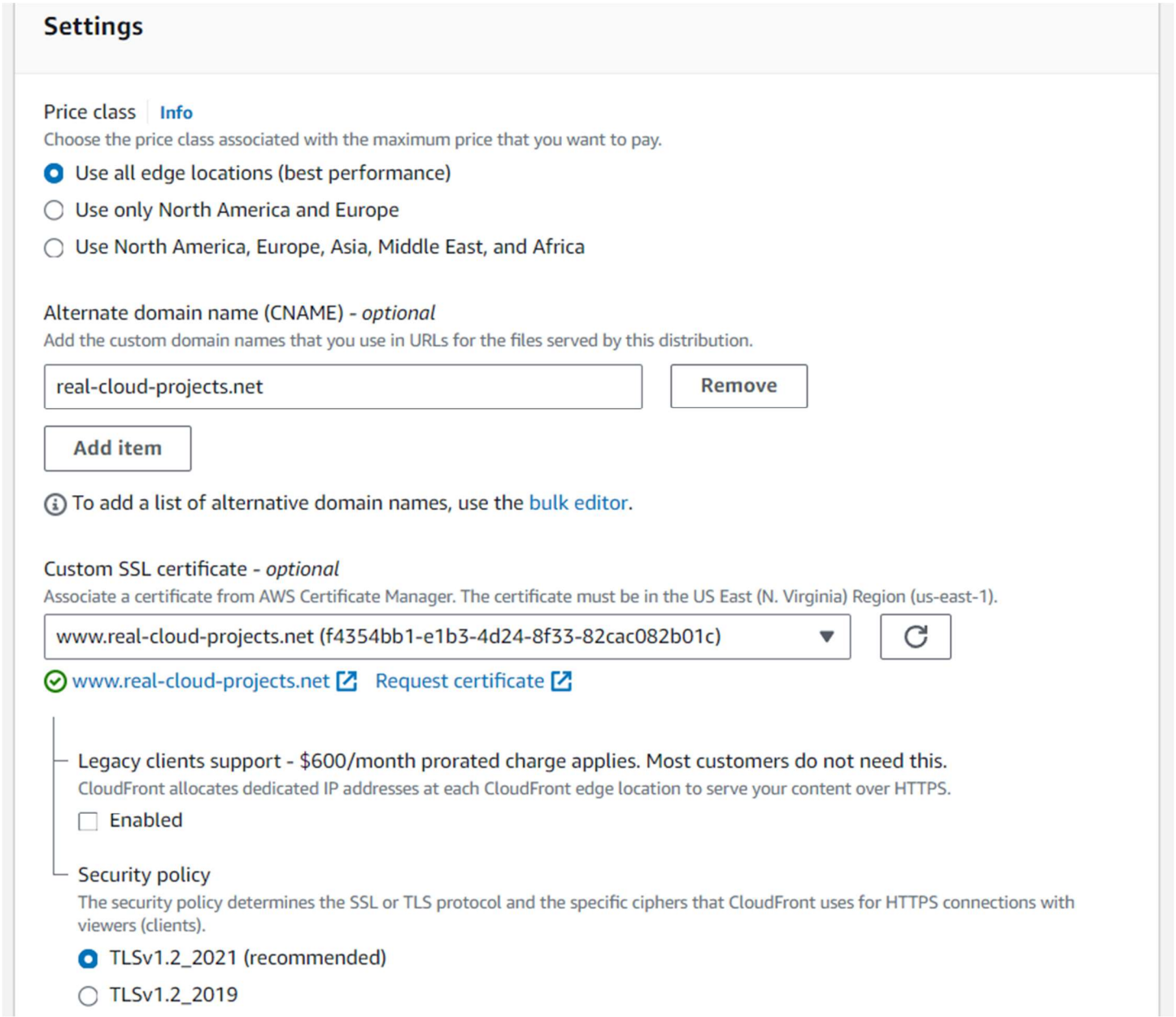
Choose an existing cache policy or create a new one.

CachingOptimized — Recommended for S3

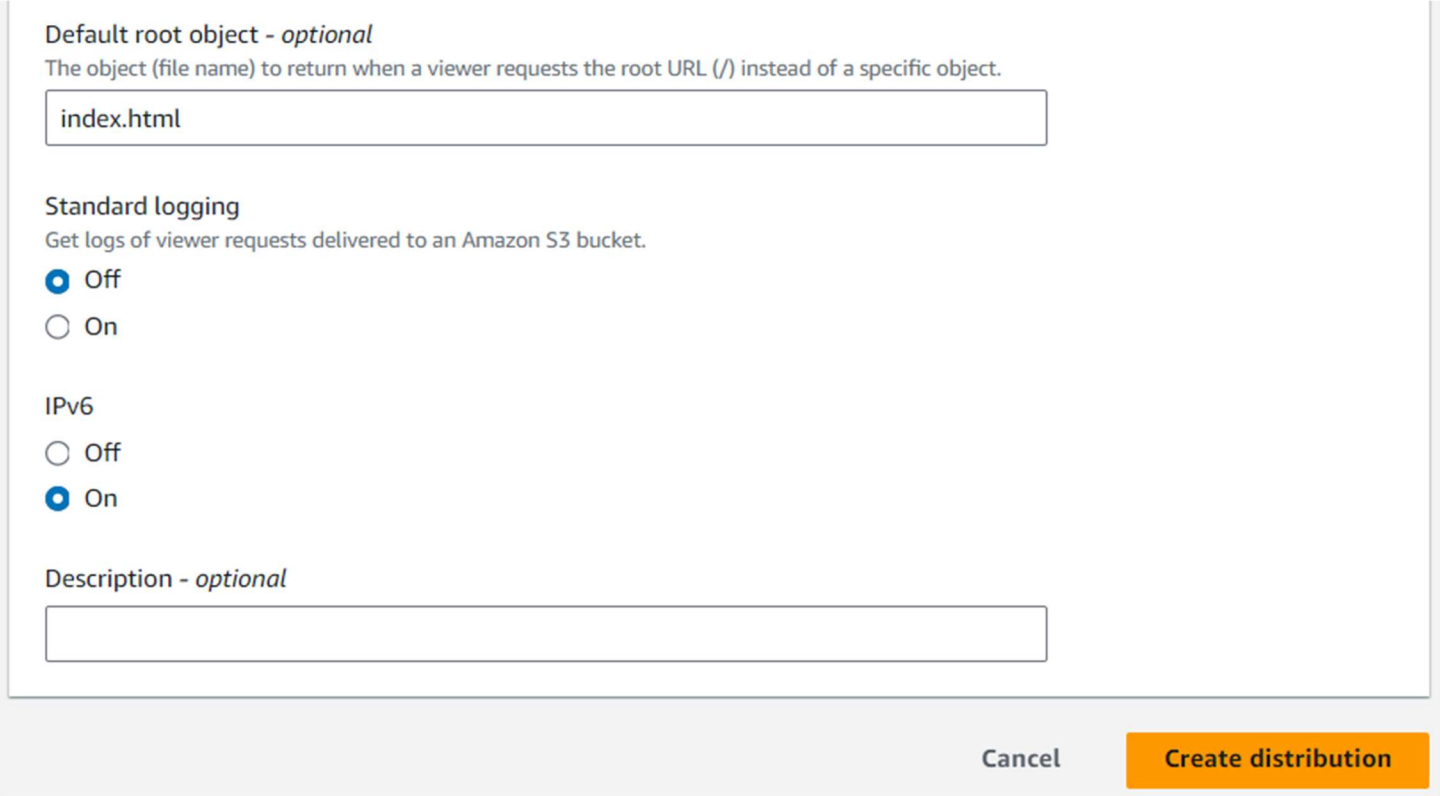
Policy with caching enabled. Supports Gzip and Brotli compression.

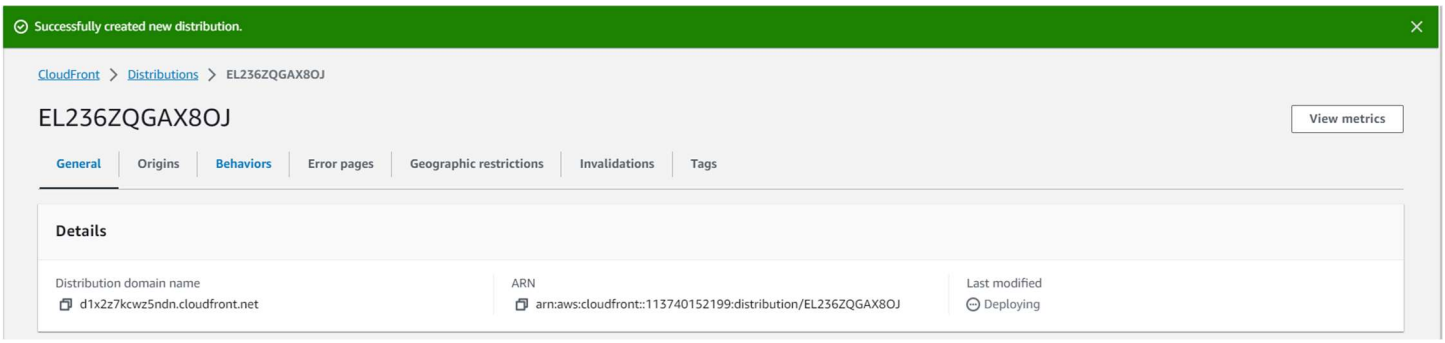
Create cache policy | View policy

Origin request policy - *optional*

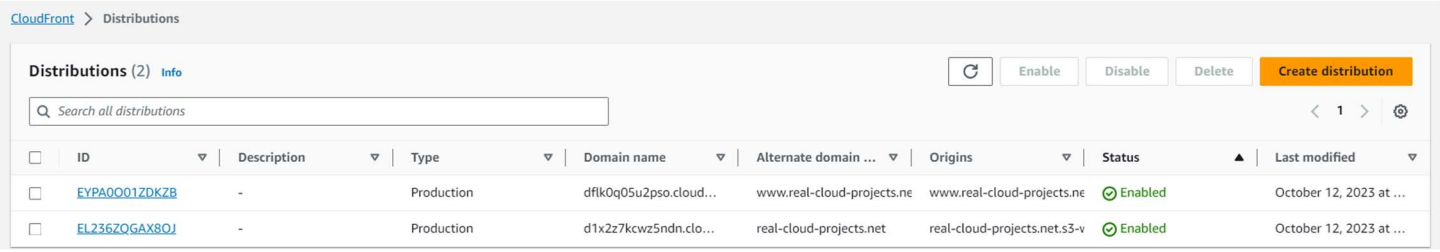


The distribution took about 10 minutes to be provisioned with Status = Enabled.



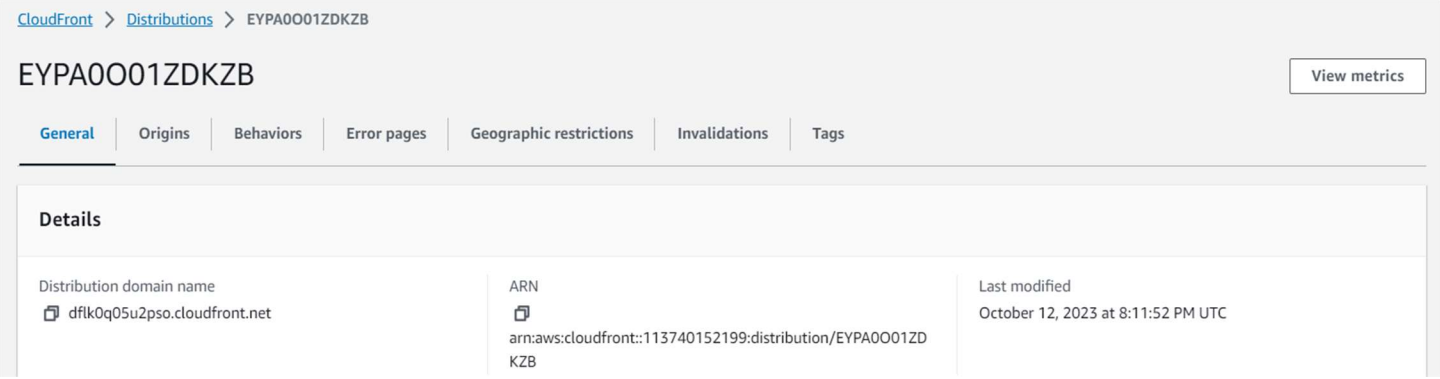


Below are the two **CloudFront** distributions that were successfully created

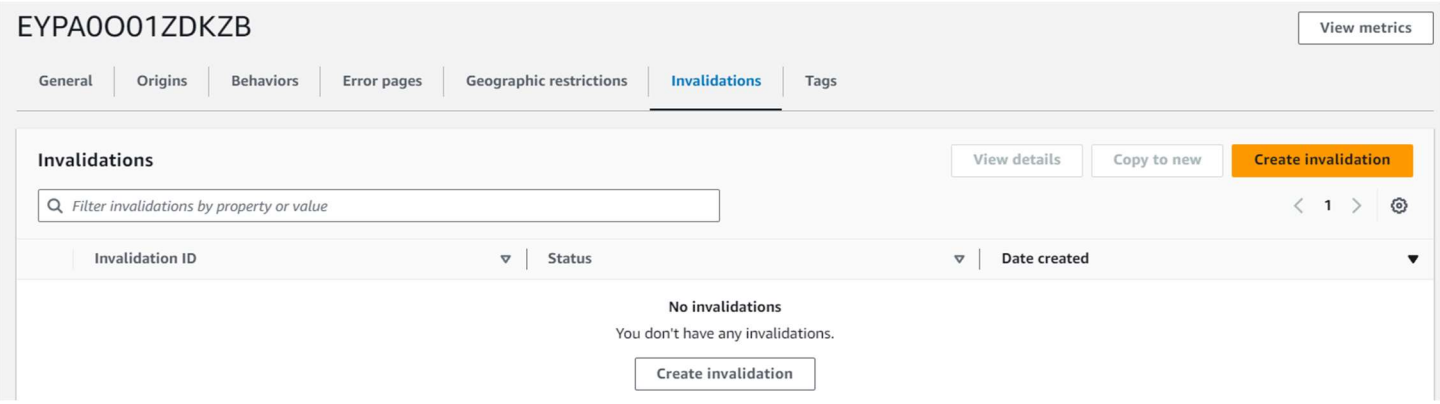


STEP 11: Configure cache invalidation of CloudFront.

CloudFront serves contents from edge locations, so that when an end-user makes another request, it can be served directly from the edge locations nearest to the user, and not from the origin S3 bucket.
Let's start by clicking on the CloudFront distribution **EYPA0O01ZDKZB**

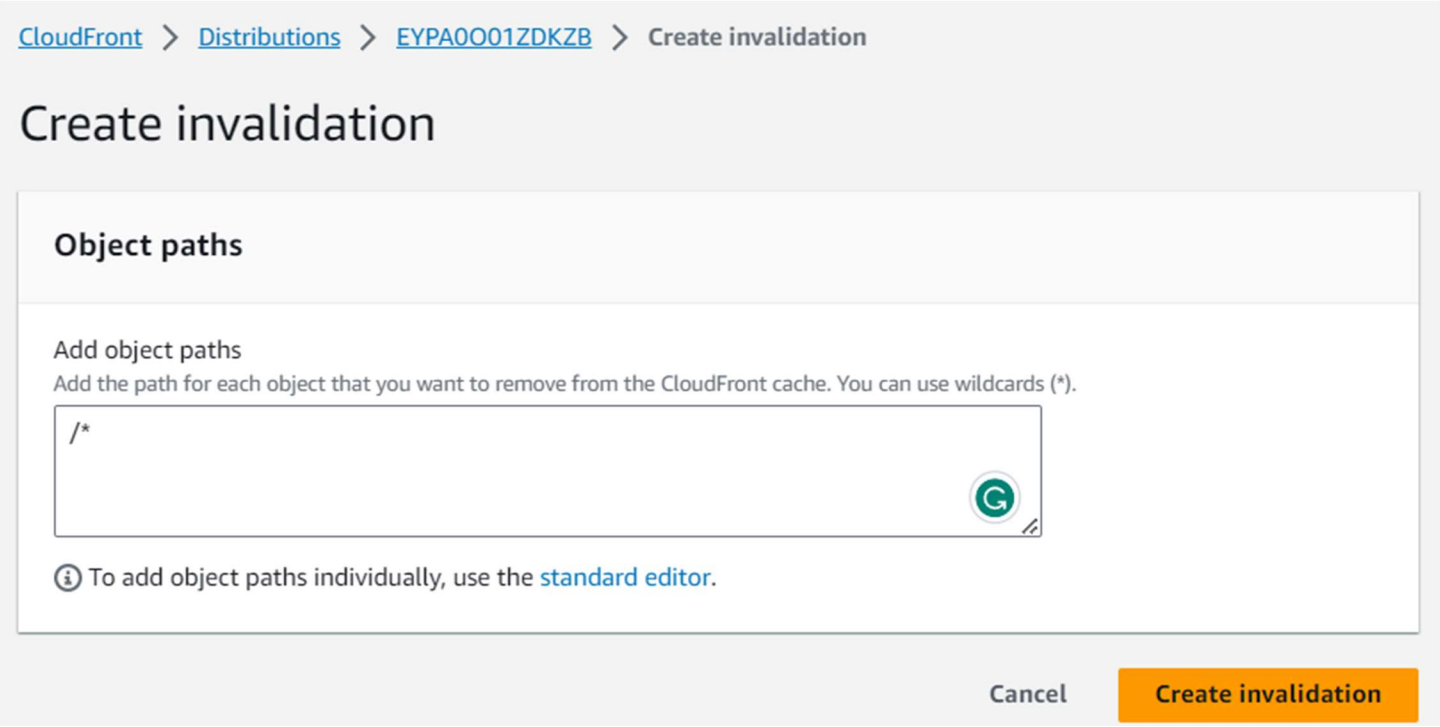


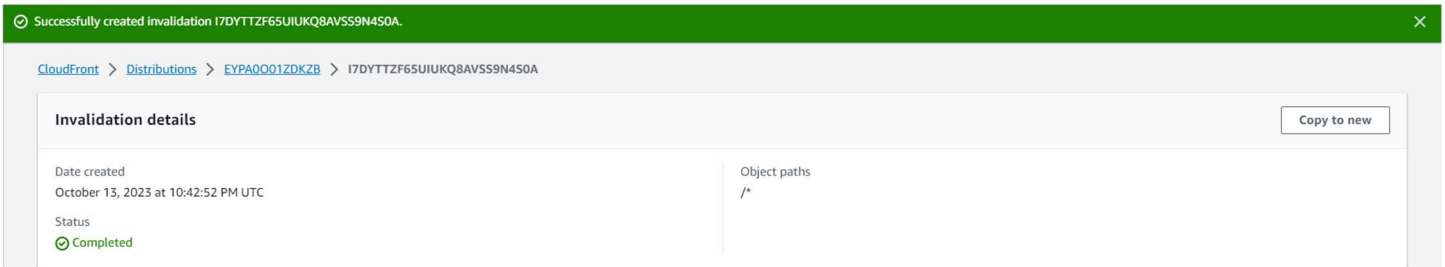
Click on "invalidations"



If any update is made on the origin, such updates can only take effect after the TTL (Time To Live) expires
Suppose I updated my website, and I want users to access the newly updated content and not the old/previous content.

Then, I will have to Invalidate all the cache, that is requesting all edge locations to update the old contents, with the newly updated content.
For this that we will:
click on **CloudFront distribution**
Click on **Invalidation**
Click on **Create invalidation /***
Click on **Invalidate**.
OR
CloudFront distribution -> invalidation -> create invalidation -> folder or file or object name -> invalidate.

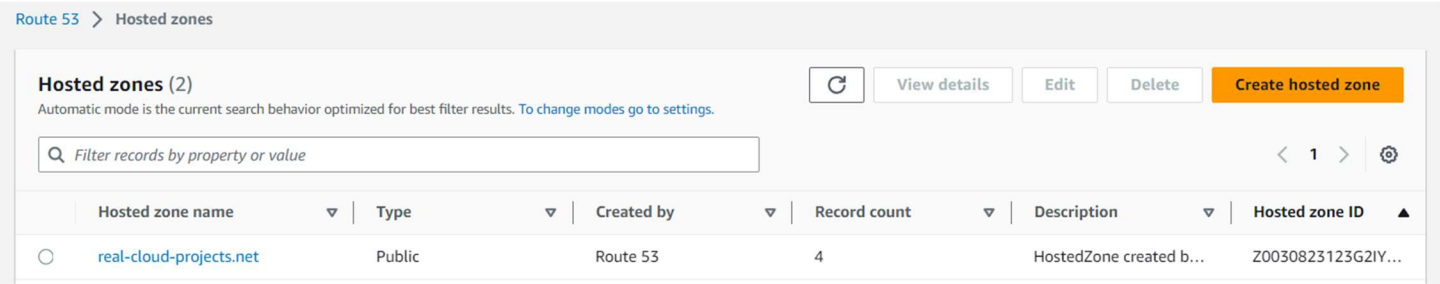


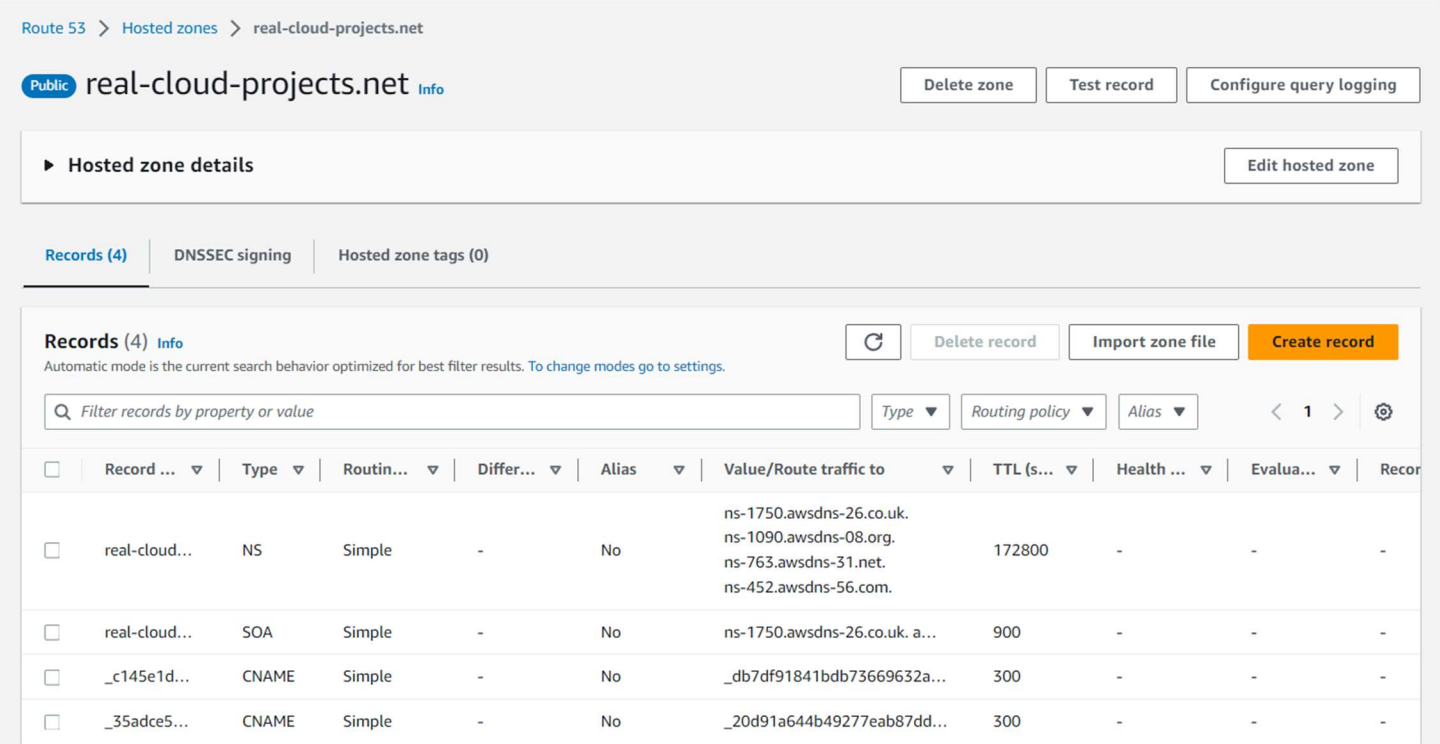


The cache was successfully Invalidated in CloudFront.

STEP 12:
Linking of CloudFront domain name with custom domain name
Navigate to the **Route 53** Dashboard

Click on **Hosted zones** **or** **domain name**
Click on **www.real-cloud-projects.net**
We already have two (2) CNAME records, one (1) NS record and two (2) SOA records





Click on **Create record**

The first record that we will create will be an A-record
Under "Record name" tab, enter www against real-cloud-projects.net

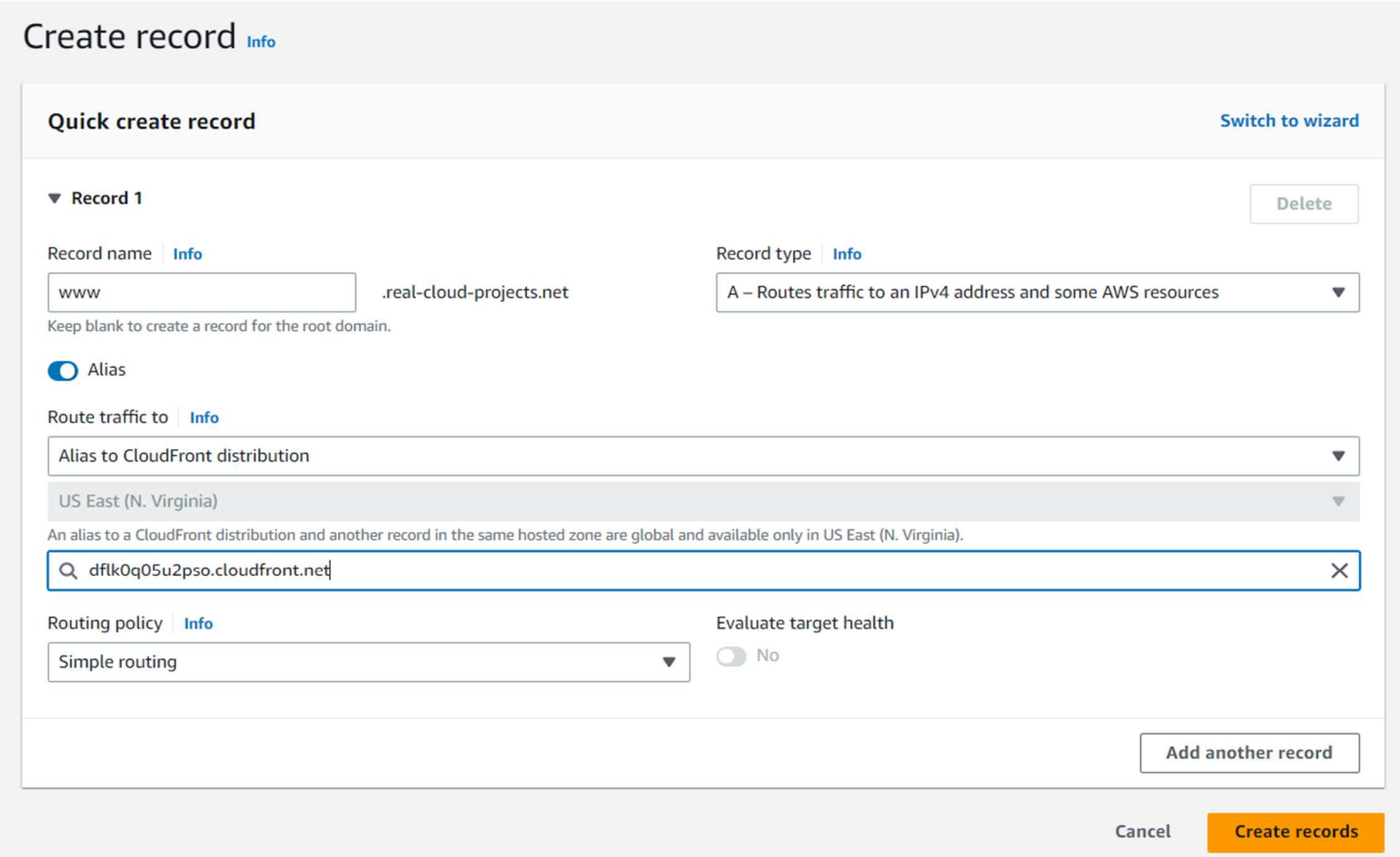
Under "Record Type" click on the dropdown
Select A-Route traffic to an IPV4 address and some AWS resources
Toggle on the Alias tab
Under "Route traffic to" click on the dropdown
Select Alias to CloudFront distribution

Region remains US east (N. Virginia)

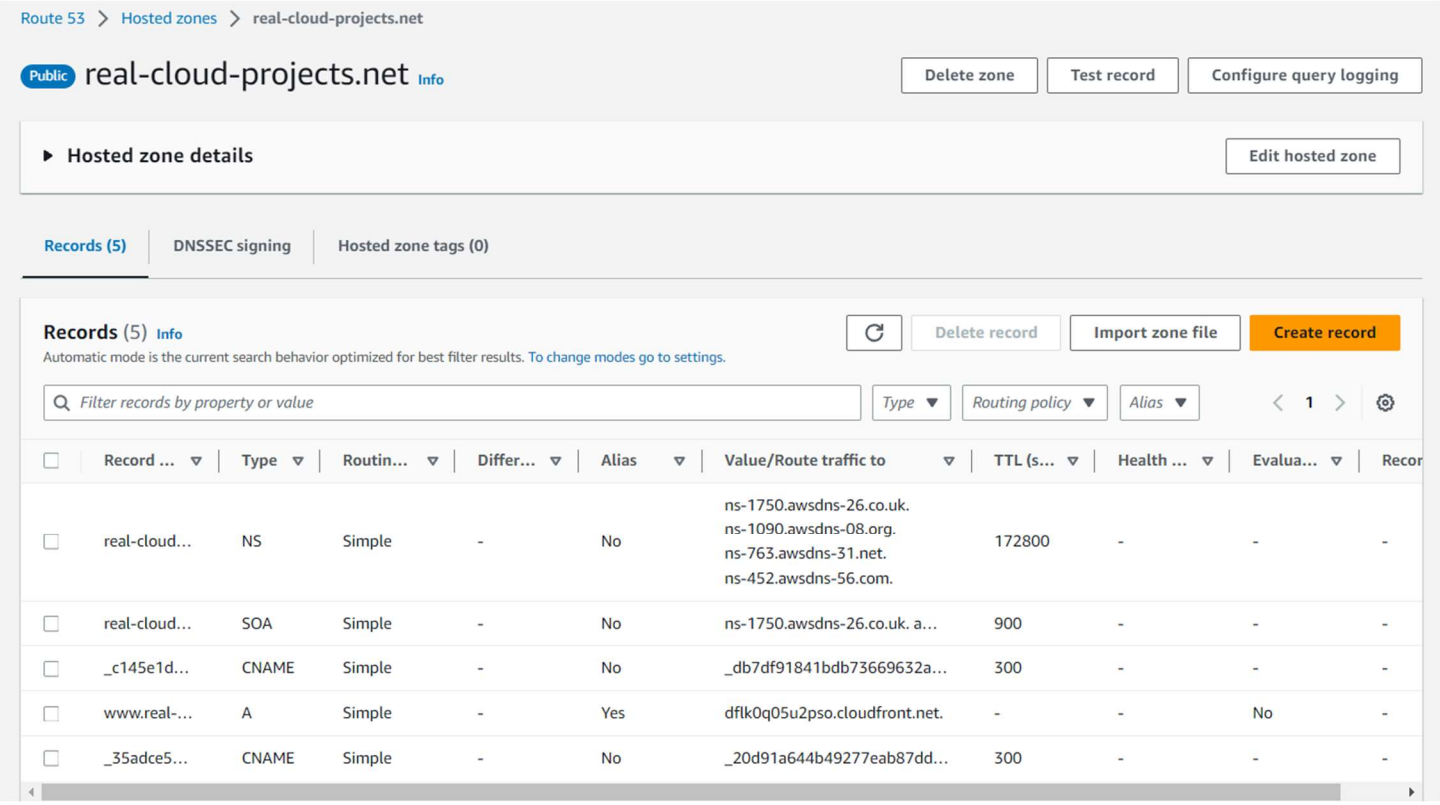
This is because the provisioned SSL Certificate issued by the AWS Certificate Manager service is only available in the US East (N. Virginia) region.

Click on the Choose distribution tab
Now, I will choose the distribution that I provisioned previously,
Select www.real-cloud-projects.net (dflk0q05u2pso.cloudfront.net)

Under "Routing policy" click on dropdown
Select Simple routing

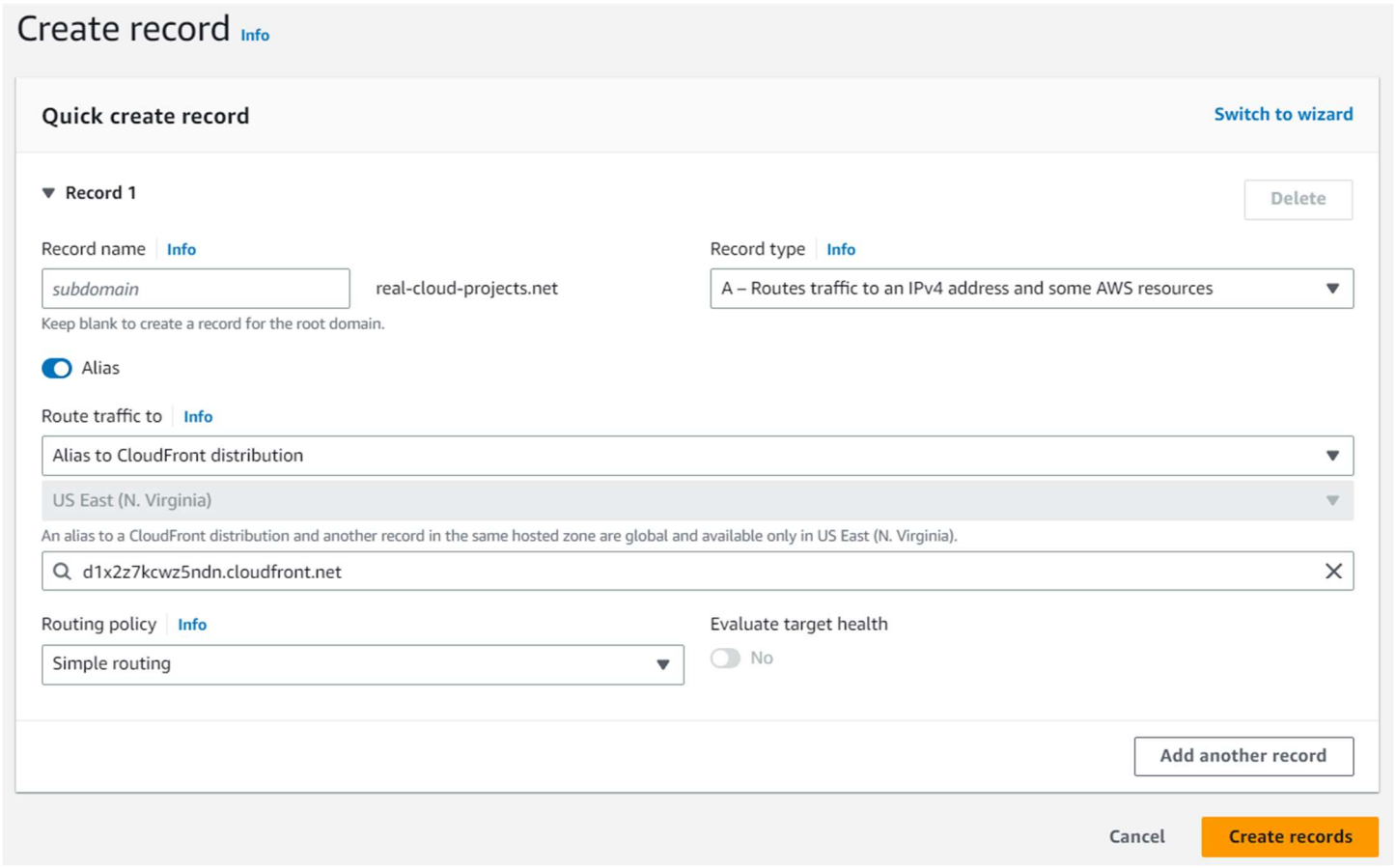


Click **Create Records**

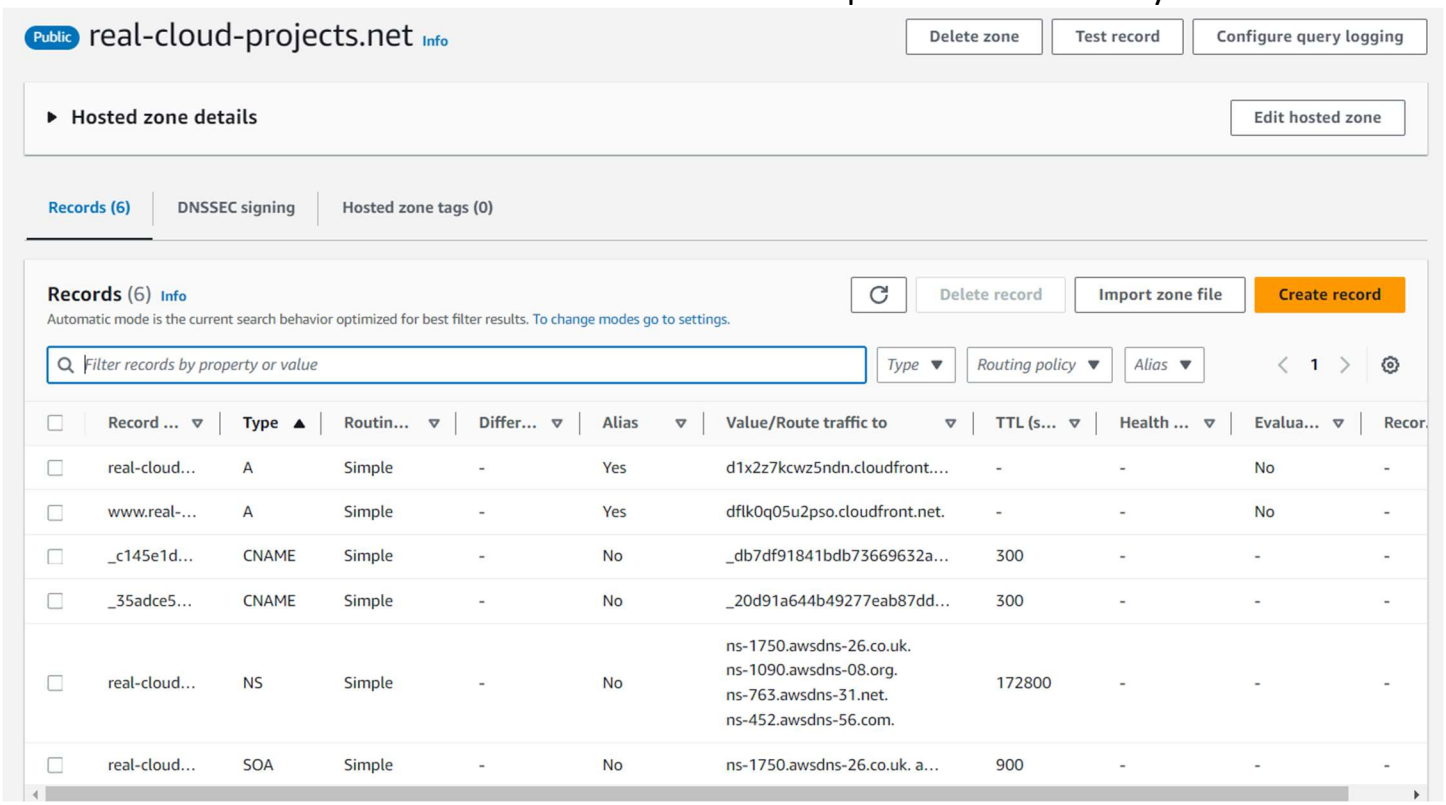


The first **A-record** for www.real-cloud-projects.net was successfully created.

Repeat the above same process with "**real-cloud-projects.net**" and corresponding public certificate to create a second Alias Record

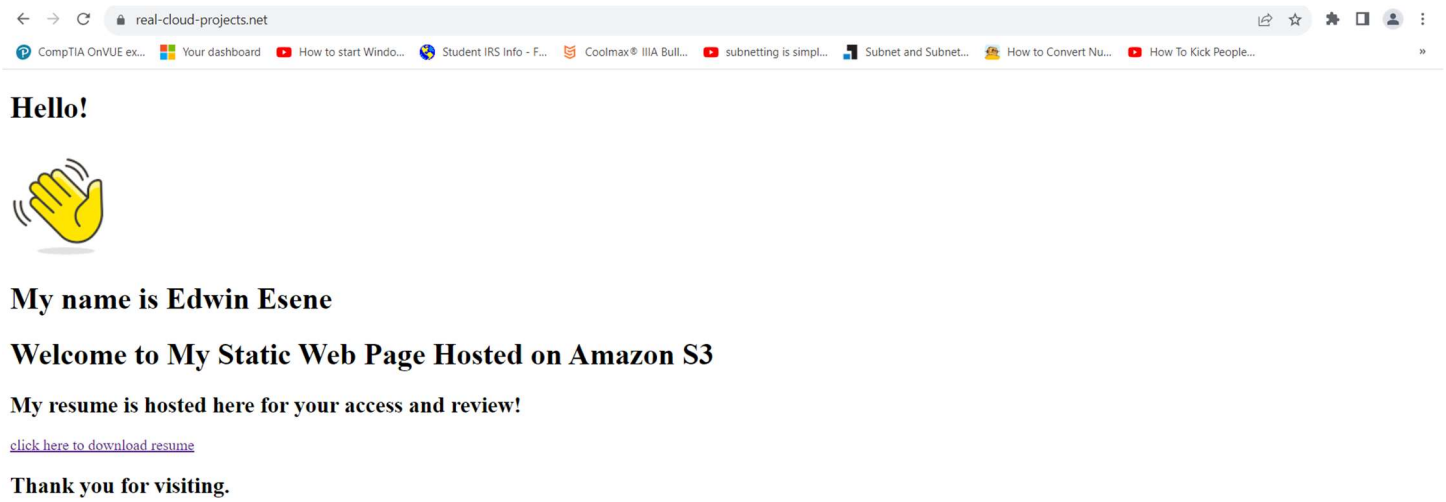


We need to wait a few minutes for the DNS records to update and to load my domain name.

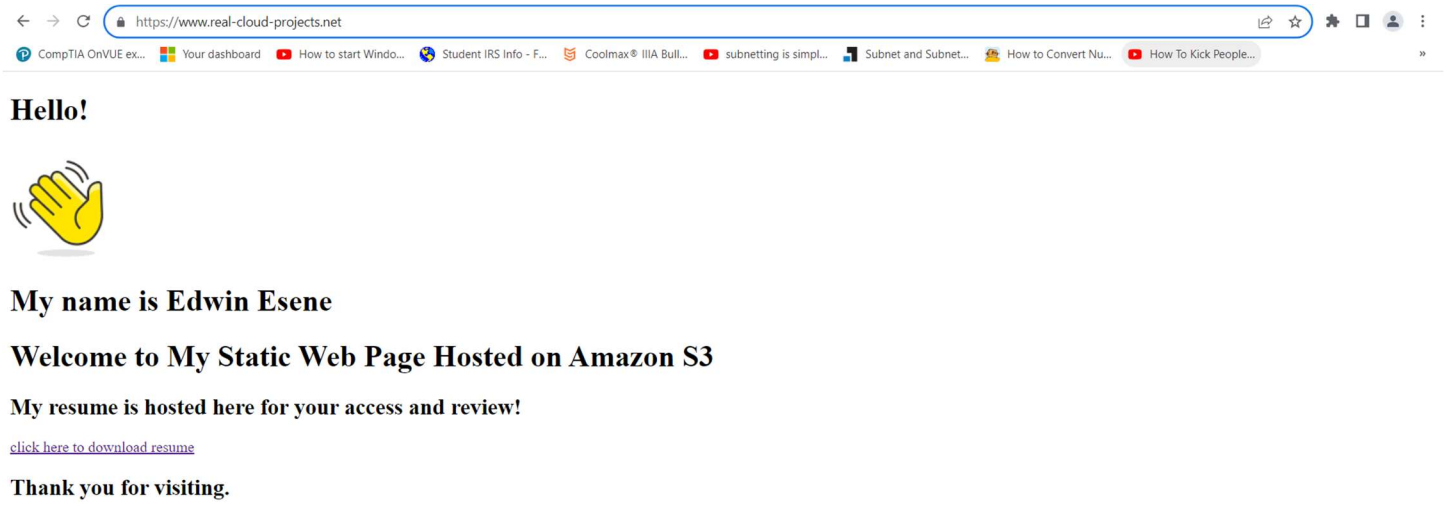


The second **A-record** for real-cloud-projects.net was successfully created as well.

Now, we have successfully created two A records which are both Alias records.
To test the solution if everything is working properly, let's copy my domain name **real-cloud-projects.net** and paste it in on a web browser.



Now I can see my static website securely distributed via CloudFront, and by double-clicking on the URL, I can see that my website has been securely redirected to **www.real-cloud-projects.net** via CloudFront as well, as shown by the screenshot URL below.



From the experiment, we can see that everything is working perfectly.

Objective of this Project

You want to serve a static page through Amazon Simple Storage Service (S3) that can be cached at a Content Delivery Location for cost optimization. My first task will be to create an S3 bucket to host a static website that will be used to serve end-users through an AWS content delivery network such as CloudFront endpoint.

What exactly is a Static Website?
Basically, a static website is made of "*fixed code*", and unless the website developer decides to execute some changes, nothing will change on the web page. Think of it like a textbook written to teach a course in Python. Static websites give a lot of the same type of information that you could get from a textbook that cannot just change by itself.

Benefits of using S3 to serve a static website.
Amazon S3 is the most fundamental and global Infrastructure as a Service (IaaS) solution provided.
Using Amazon S3 facilitates highly scalable, secured, and low-latency data storage from the cloud. With its simple web service interface, it is easy to store and retrieve data on Amazon S3 from anywhere on the global network. Amazon S3 is a pioneer in cloud data storage with various benefits, some of which are listed below:

(1) **Reliable Security:** By providing access permissions, you can ensure that there is no unauthorized access to the data stored in an S3 bucket.

(2) **Cost Effective:** With Amazon S3, you only pay for the data you use. Hosting a small to medium sized static website would only cost a few dollars monthly.

(3) **Simplicity of Management:** Amazon S3 has a very user-friendly web interface that enhances efficiency in terms of maintaining security, optimizing storage classes, and managing data transfer in the most suitable way. With everything in place, your website can be configured and up running in a few minutes.

What is CloudFront?
Amazon CloudFront is a content delivery network (CDN) service built for high performance, low latency, security, and convenience.
It is a data or content caching technology that resides at defined edge locations. If the content is already in the edge location with the lowest latency, CloudFront delivers it immediately. But, If the content is not in that edge location, CloudFront will retrieve it from an origin that you've defined, such as an Amazon S3 bucket, that you have identified as the source for the definitive version of your content.

Benefits of using CloudFront
(1) **Global Scaled Network for Fast Content Delivery:** Amazon CloudFront global edge network currently consists of 410+ points of presence (PoPs), including more than 400 Edge Locations, and 13 regional mid-tier caches in over 90 cities across 48 countries that are interconnected via the AWS backbone delivering ultra-low latency performance and high availability to your end-users.

(2) **Security at the edge:** All your CloudFront distributions are defended by default against the most frequently occurring network and transport layer DDoS attacks that target your websites or applications with AWS Shield Standard.

(3) **Deep integration with AWS:** Amazon CloudFront is integrated with AWS services such as Amazon S3, Amazon EC2, Elastic Load Balancing, Amazon Route 53, and AWS Elemental Media Services for easy set-up.

(4) **Cost-Effective:** Amazon CloudFront offers cost-effective content delivery globally. Integrated with AWS, there are no transfer fees for origin fetches from any AWS origin.

(5) **AWS Certificate Manager** (ACM) offers custom TLS certificates, at no charge.

Ladies and Gentlemen, we have reached the end of this impressive project and I hope you liked it. See you soon on the next project.