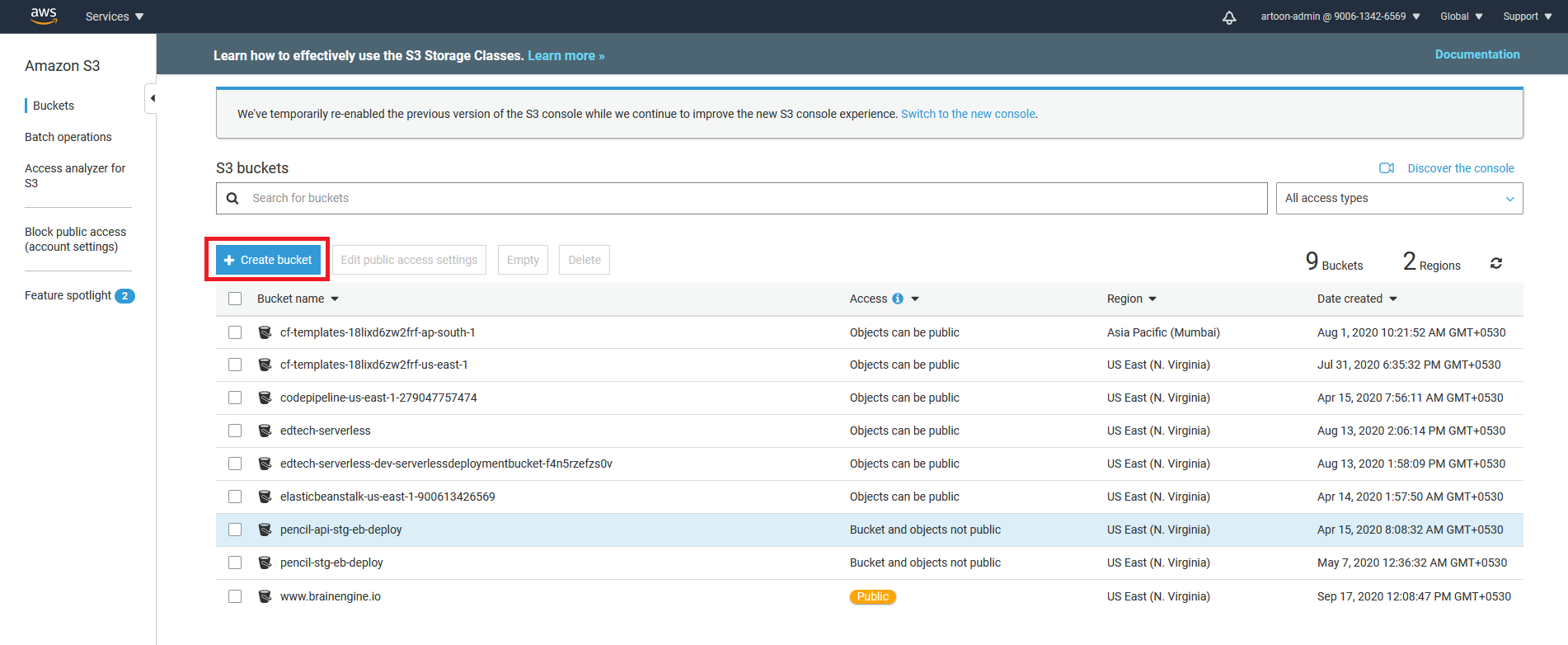
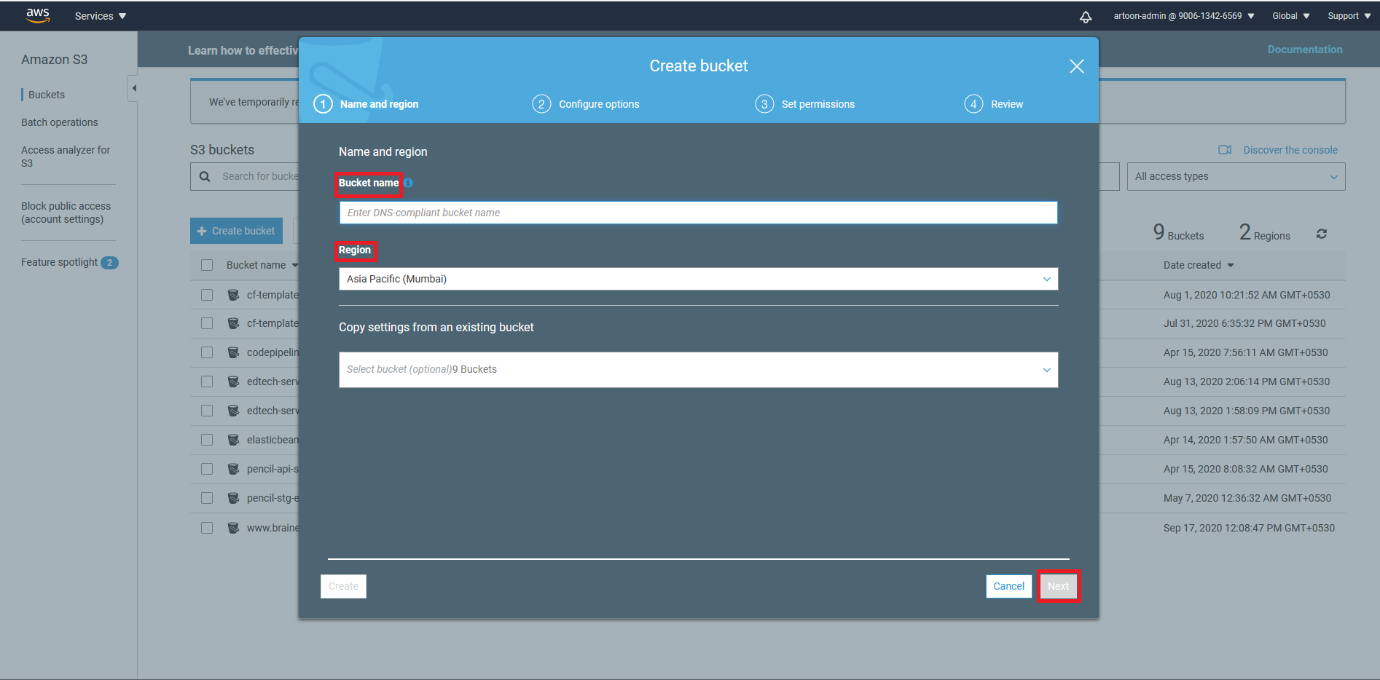
S3 Bucket Web Hosting.

1. Login to AWS account. And going to s3 bucket service
2. Create the bucket

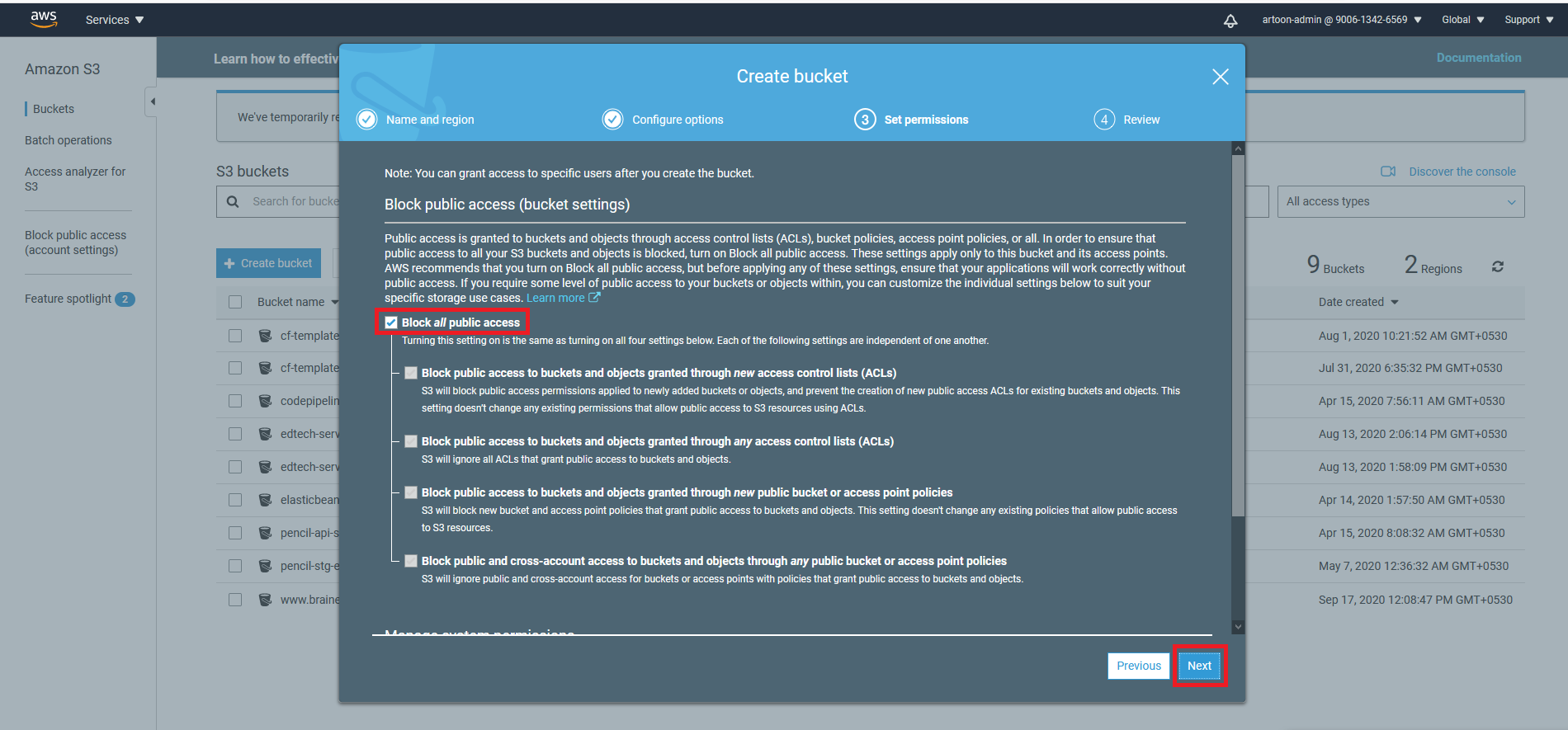


Select the create bucket button for the create new bucket.

1. Enter the bucket name and select the region then click on **NEXT** button.

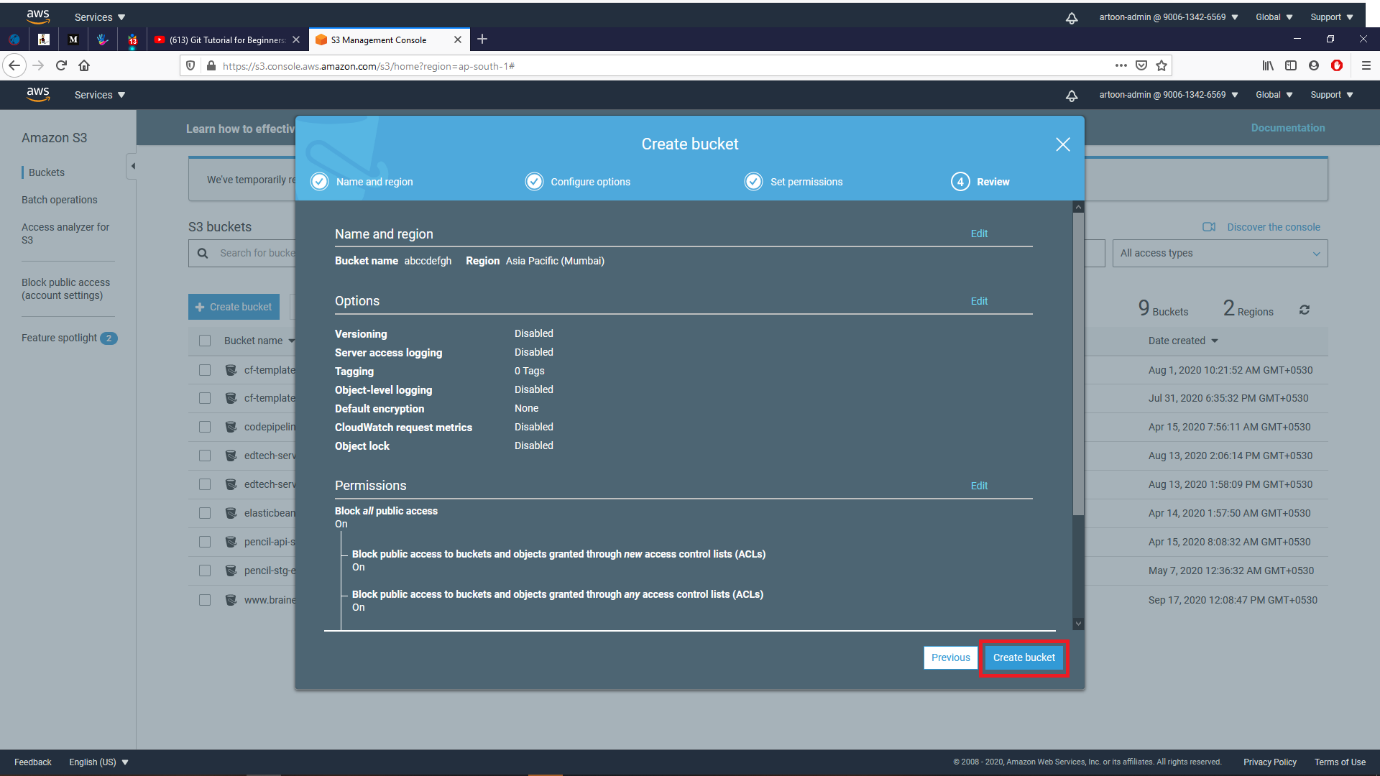


1. Set the permission for the bucket.



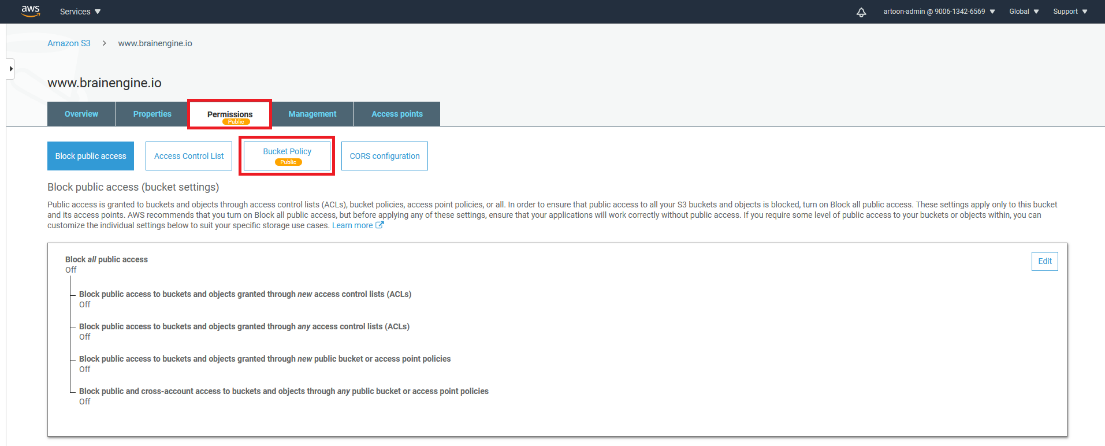
Uncheck the check box for the public access. Then click on **NEXT** button.

1. Review your bucket.

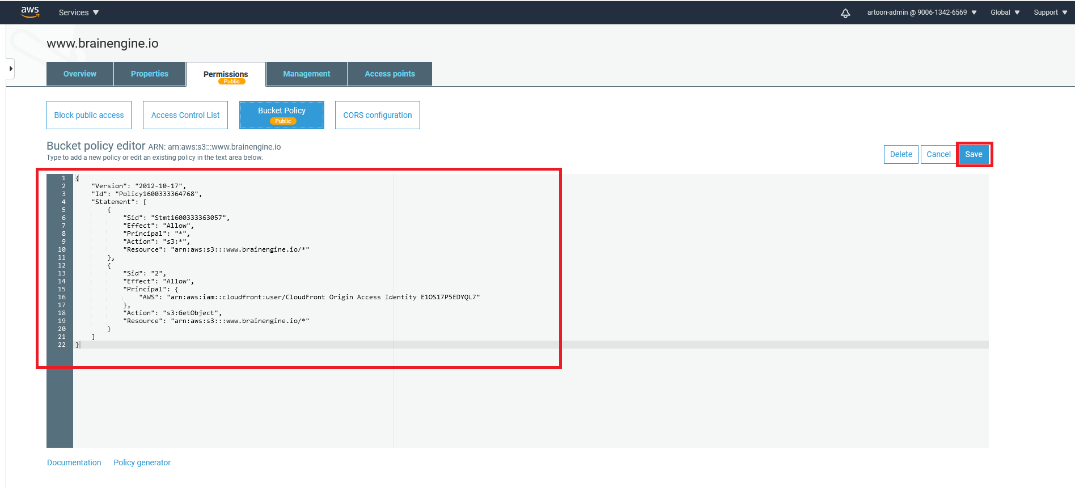


Then click on **CREATE BUCKET**  button.

1. Then enter in bucket. Then select the **PERMISSION** option.



Enter in the **BUCKET POLICY** option.



Enter the policy of the bucket for the PUBLIC access and then SAVE it.

**POLICY Example:**

{

"Version": "2012-10-17",

"Id": "Policy1600333364768",

"Statement": [

{

"Sid": "Stmt1600333363057",

"Effect": "Allow",

"Principal": "\*",

"Action": "s3:\*",

"Resource": "arn:aws:s3:::www.brainengine.io/\*"

},

{

"Sid": "2",

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access Identity E1OS17P5EDYQL7" ---**change the cloudfront distribution ID**

},

"Action": "s3:GetObject",

"Resource": "arn:aws:s3:::www.brainengine.io/\*" ---**change bucket ARN**

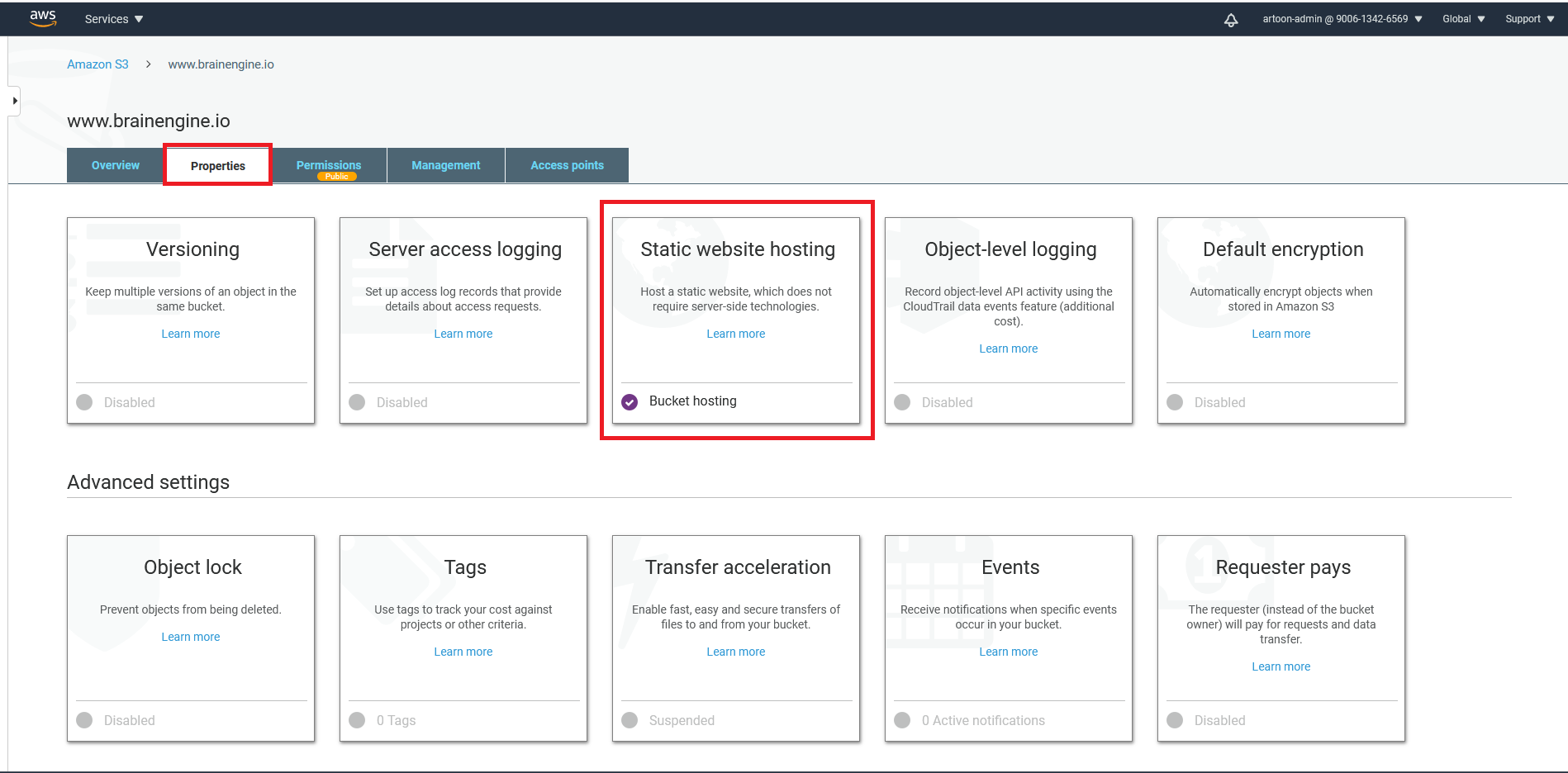
}

]

}

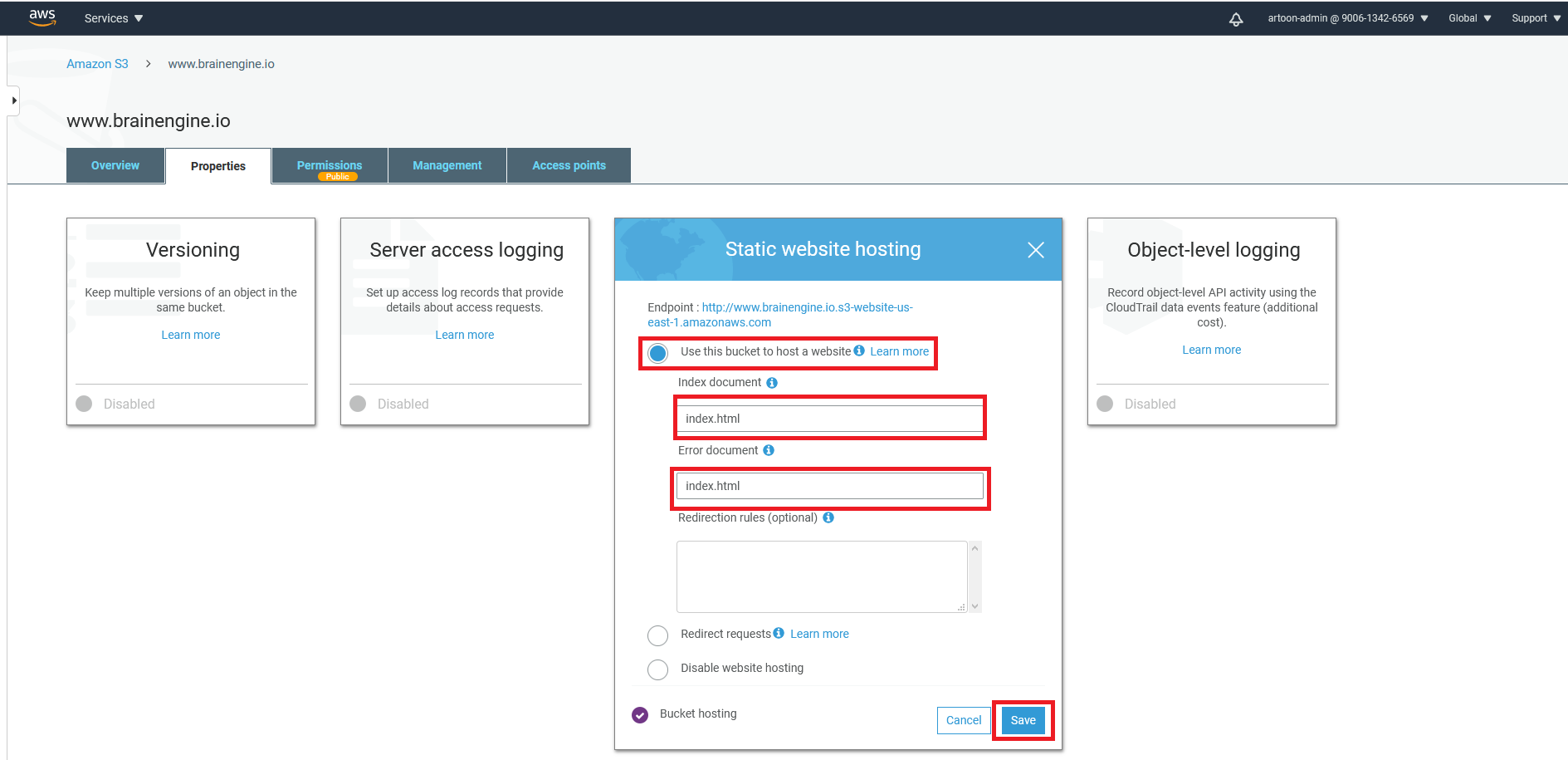
Static Web Hosting





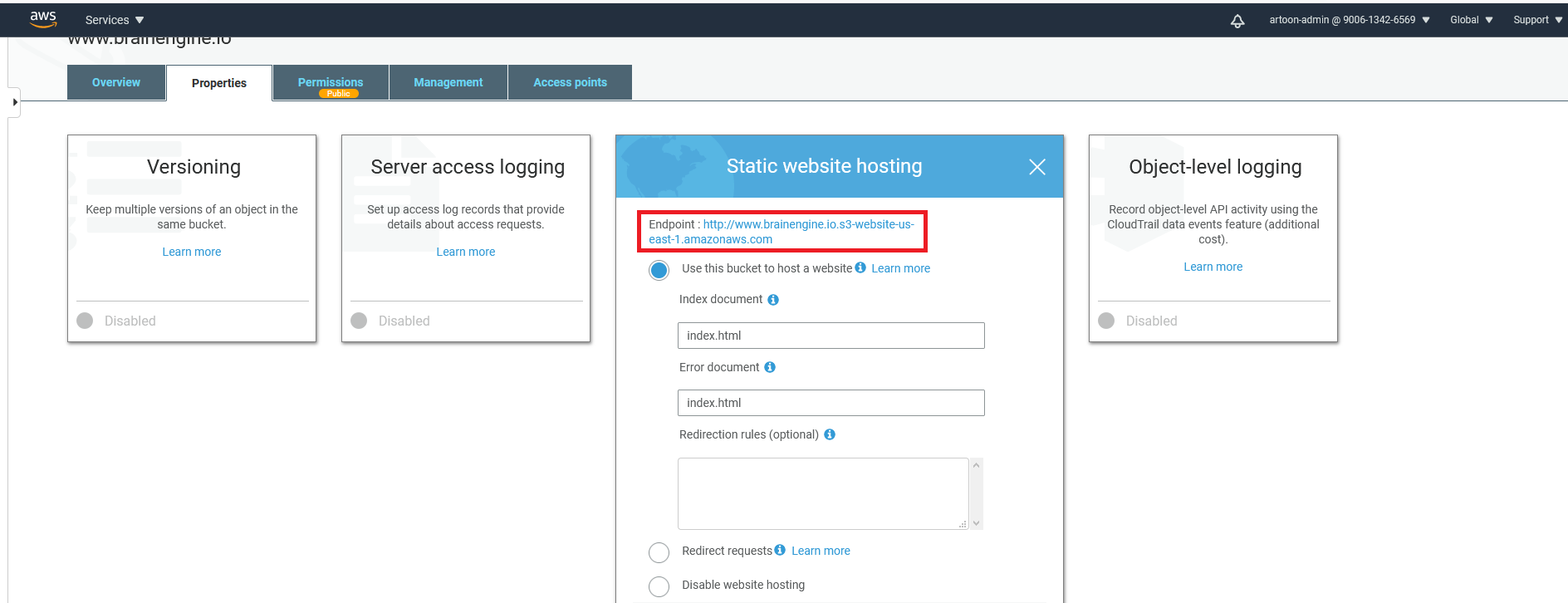
Now you can click the **Properties** menu and select the **STATIC WEBSITE HOSTING**  menu.





Then select the **Use this bucket to host a website** option select and the enter the **Index document** and  **Error document** then click on **Next**  button.

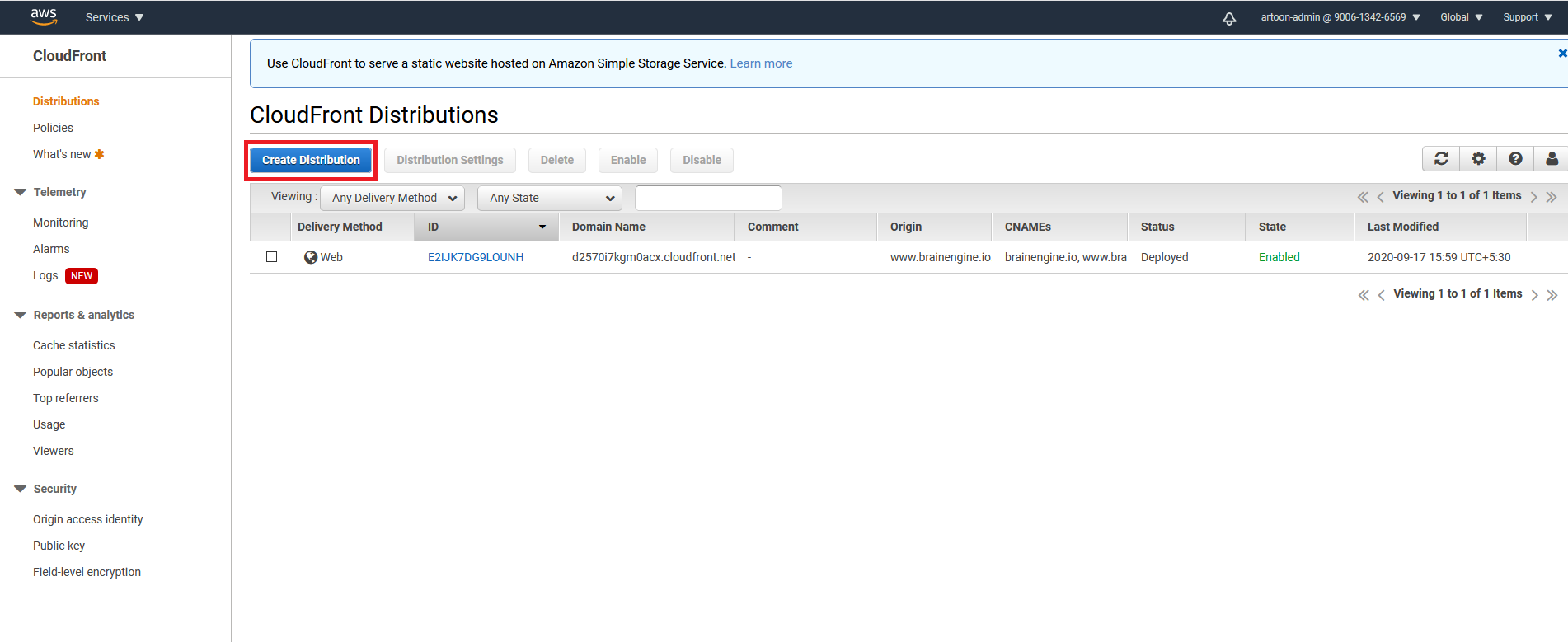




Then you can access the **Endpoint URL** for the website check.

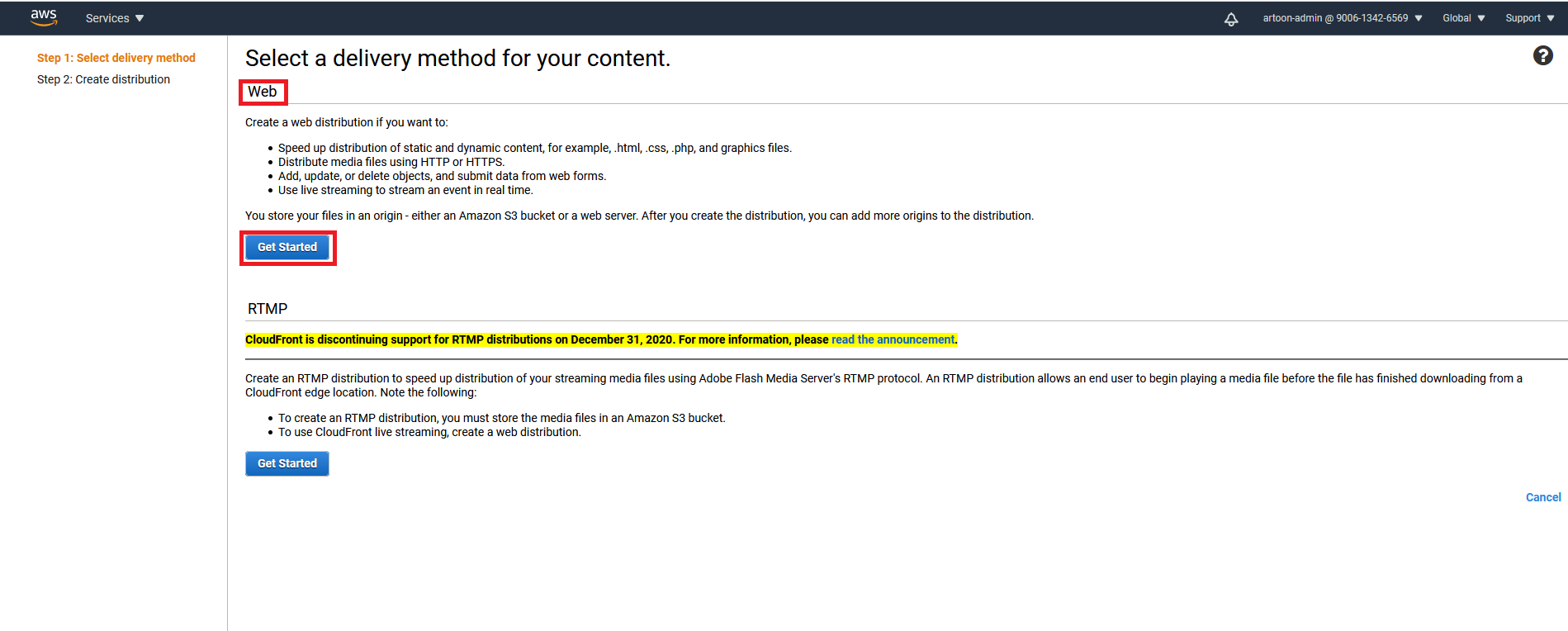
Cloud Front Distribution

1. Search in service of CloudFront and open.

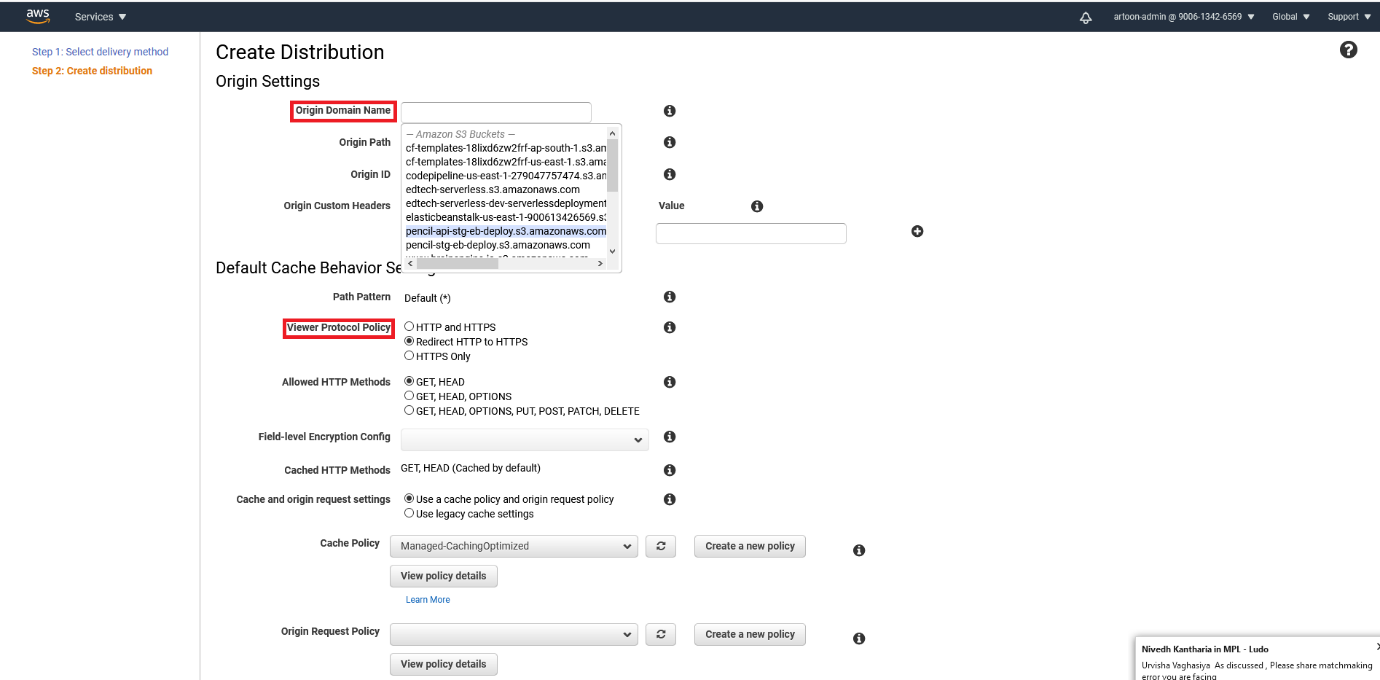


Click the **Create Distribution** button for the create.





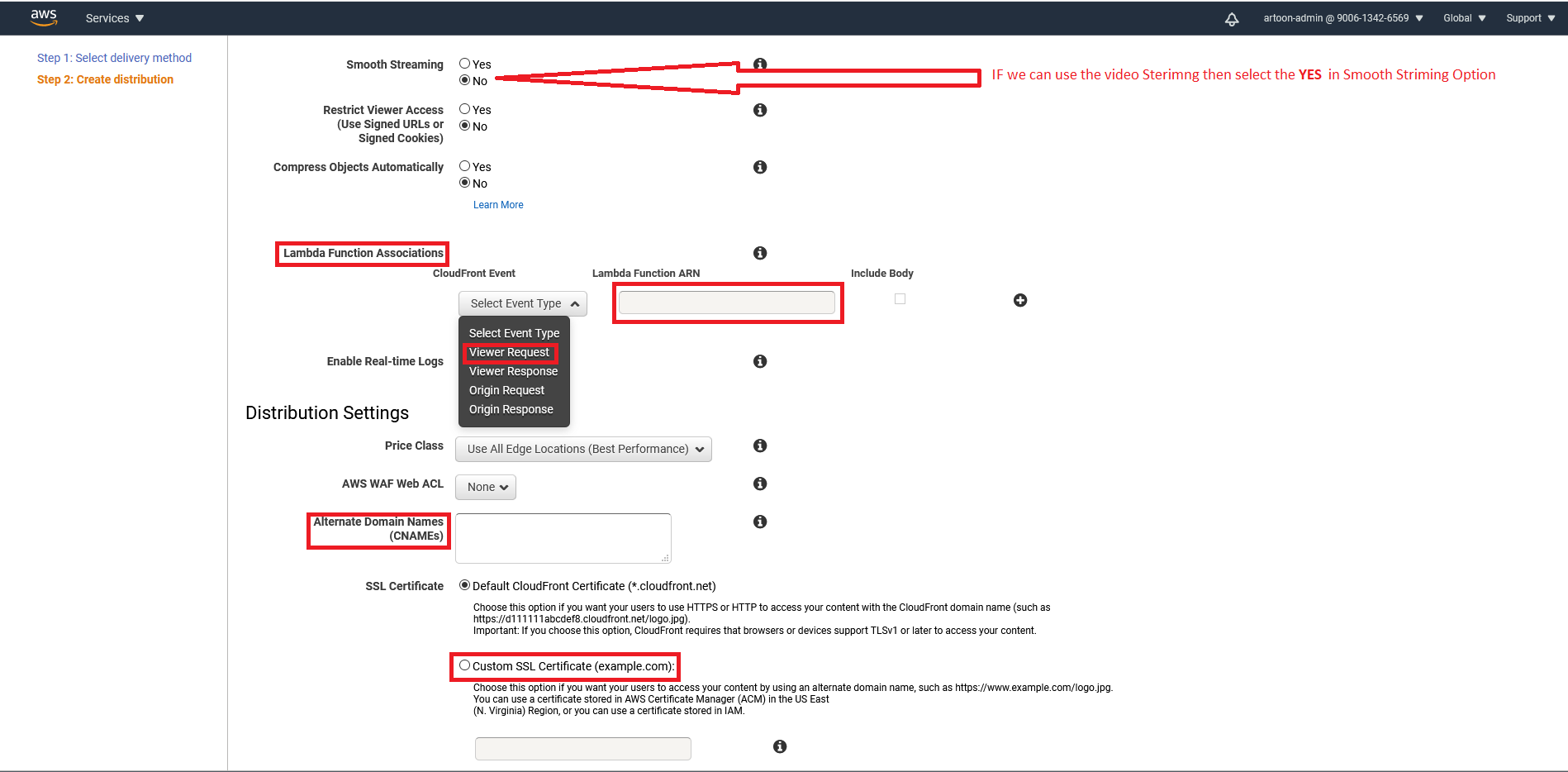
Select the delivery method and create the **WEB**  method and click the **Get Started**  Button.



Select the origin Domain Name : select your **s3 bucket**

Select the viewer protocol policy: select the **Redirect HTTP to HTTPS.**

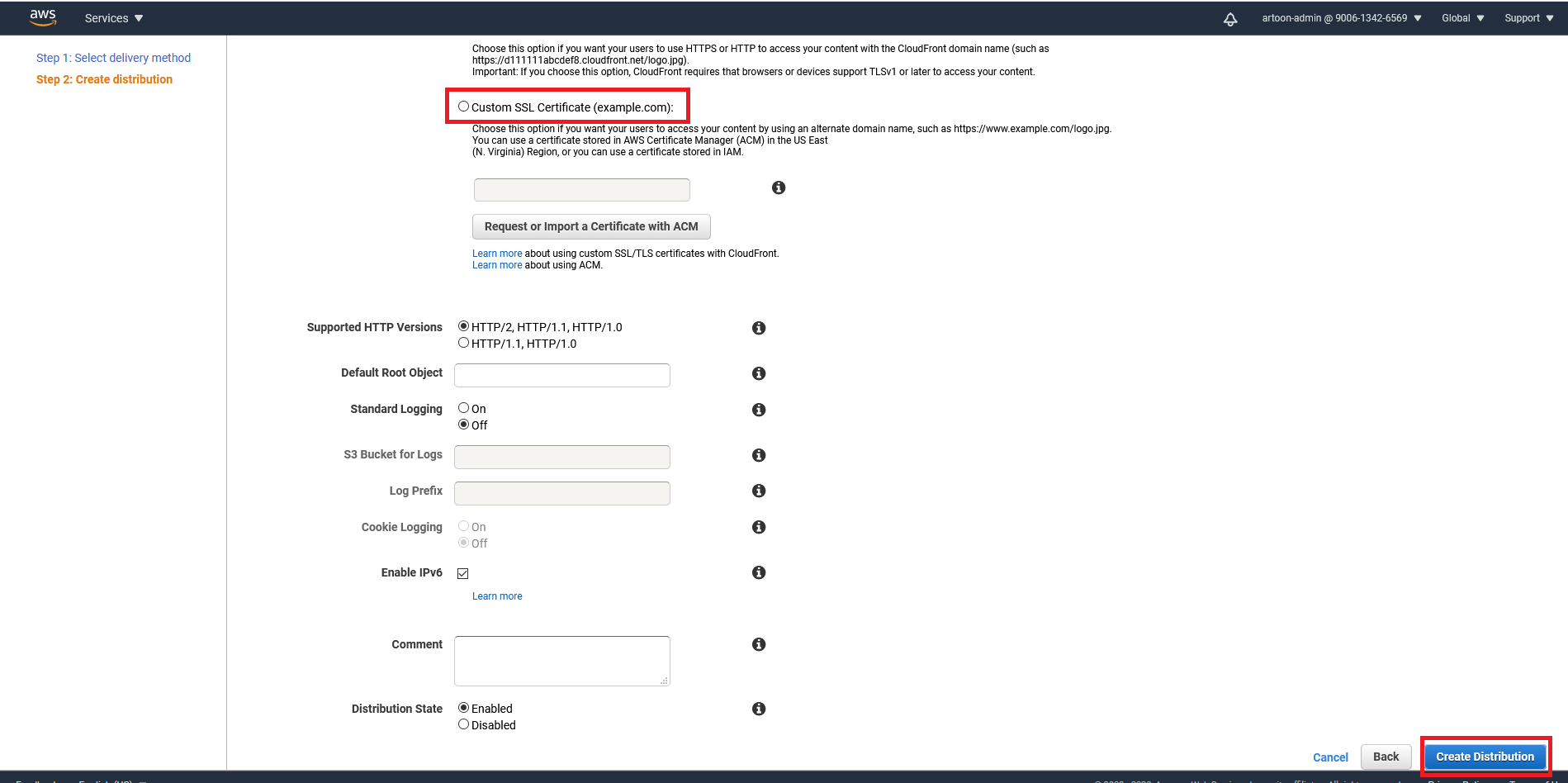




We can use the s3 bucket Authentication then select the **lambda function association** option:

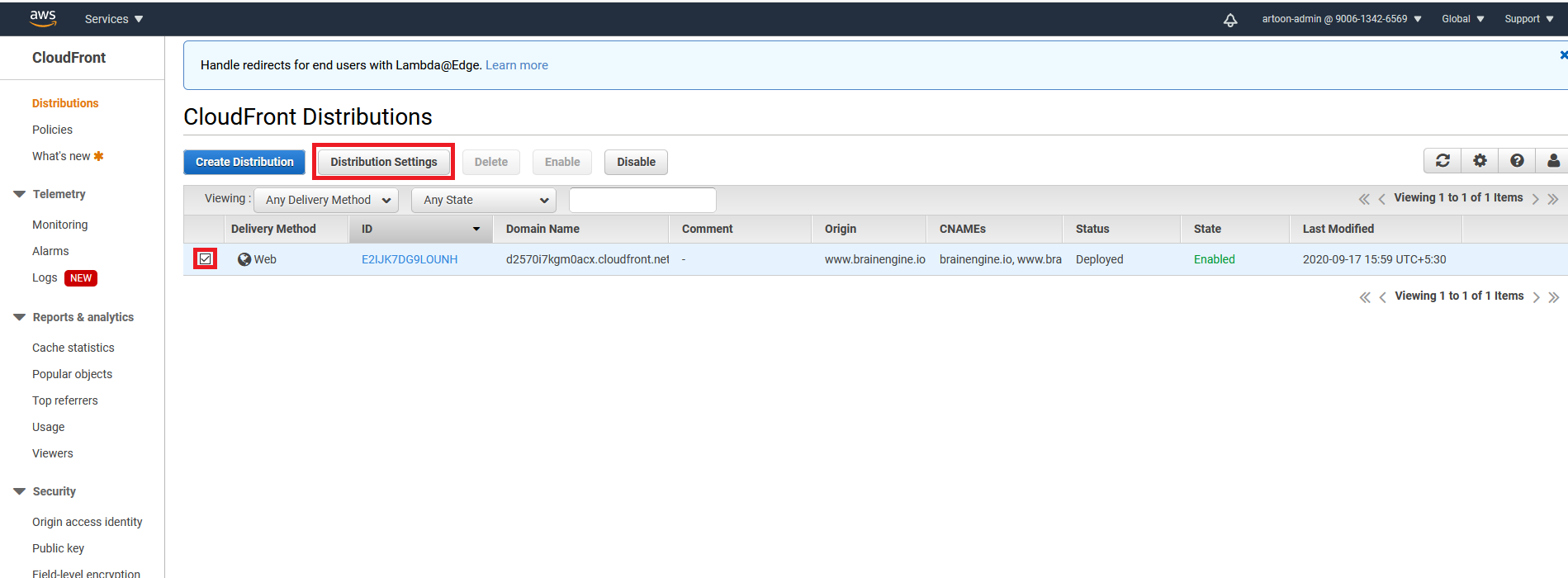
* Cloud front Event: **Viewer request** then Enter the **Lambda function ARN.**
* Alternet Domain Name : Enter the **(CNAME).**
* Choose the **Custom SSL Certificate**: select Your certificate.



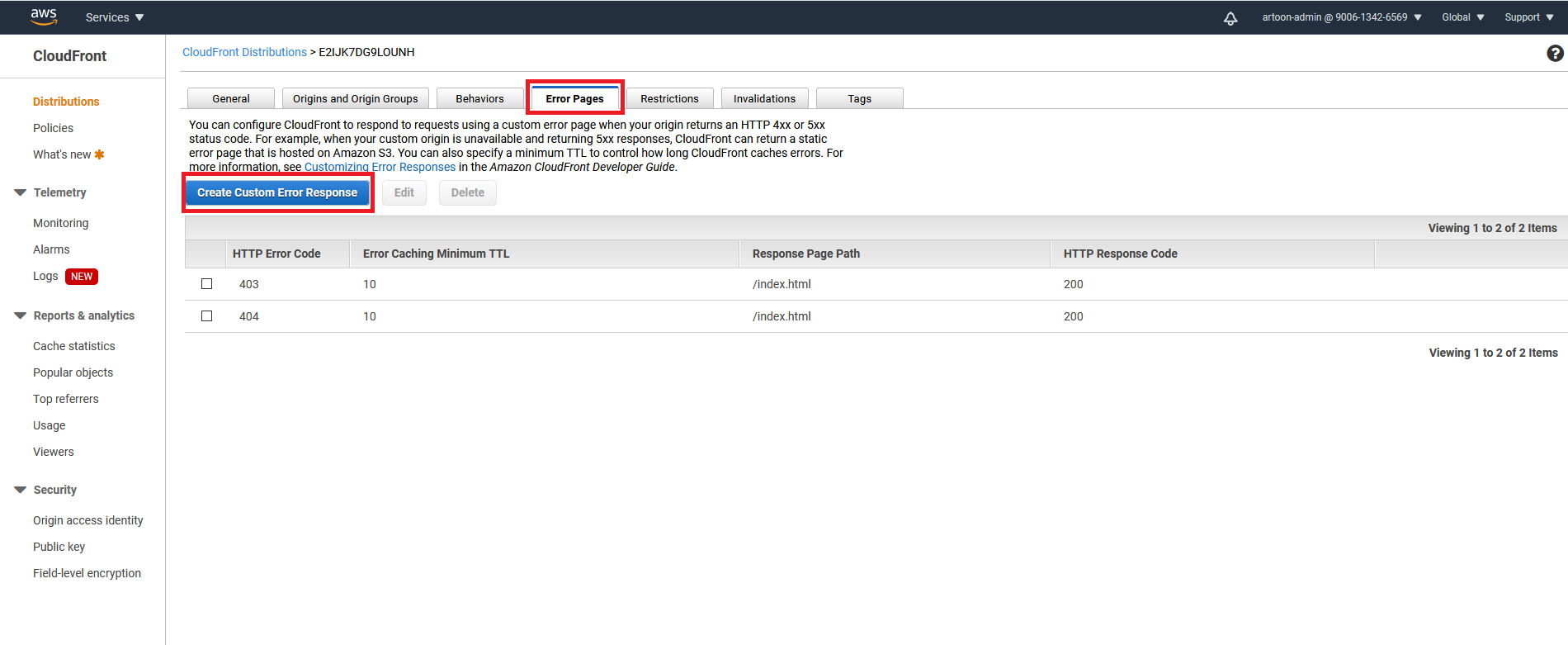


Then select the **Create Distribution** Button.

1. Add the Error Pages:



Select your Distribution and select the **Distribution Setting.**



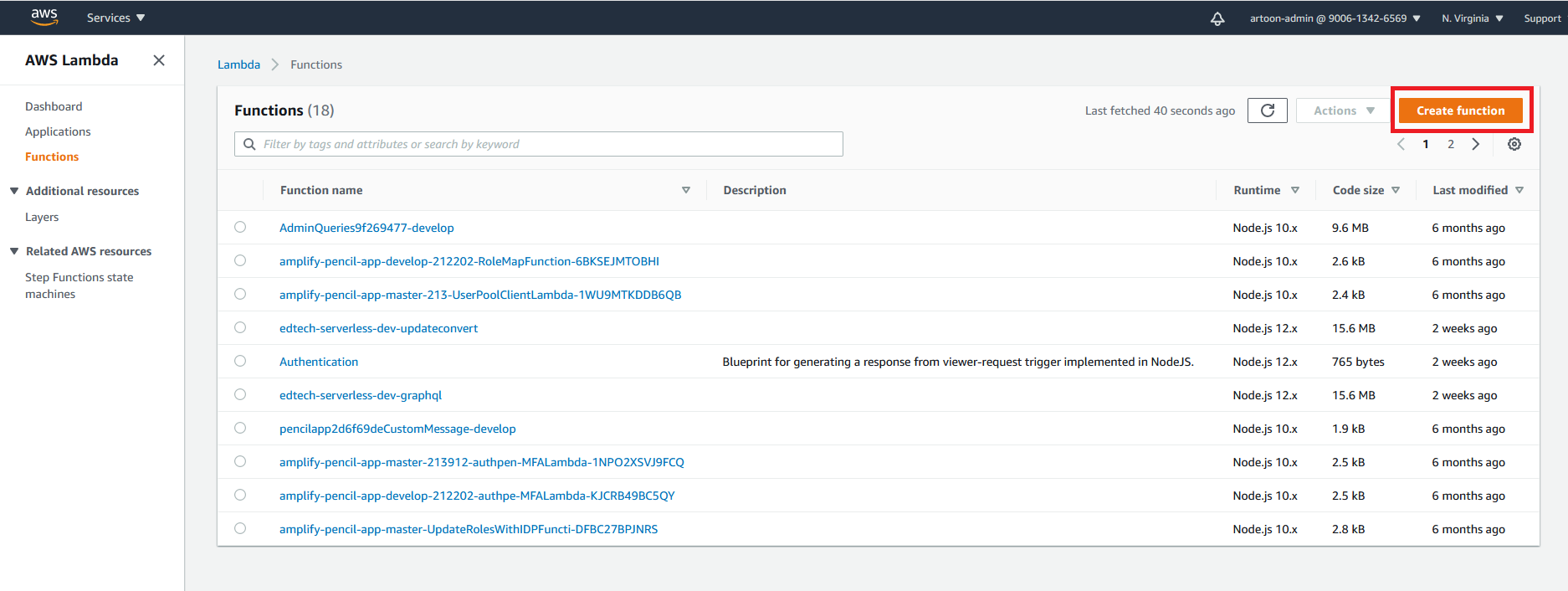
Select the **Error Pages** Tab.  
create the custom error response.



Create like this **Custom Error Response** .

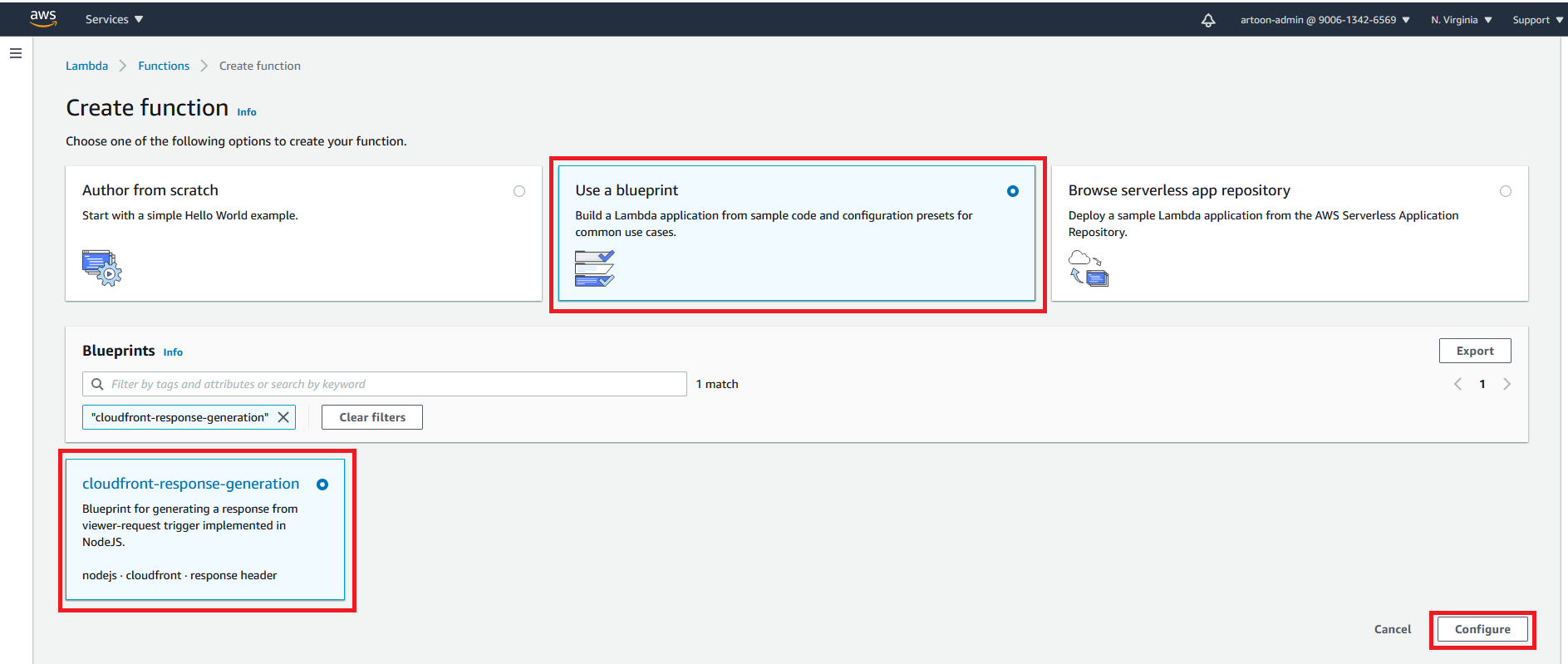
Create Lambda Function For S3 Bucket Authentication.

1. Search the service of Lambda and open it



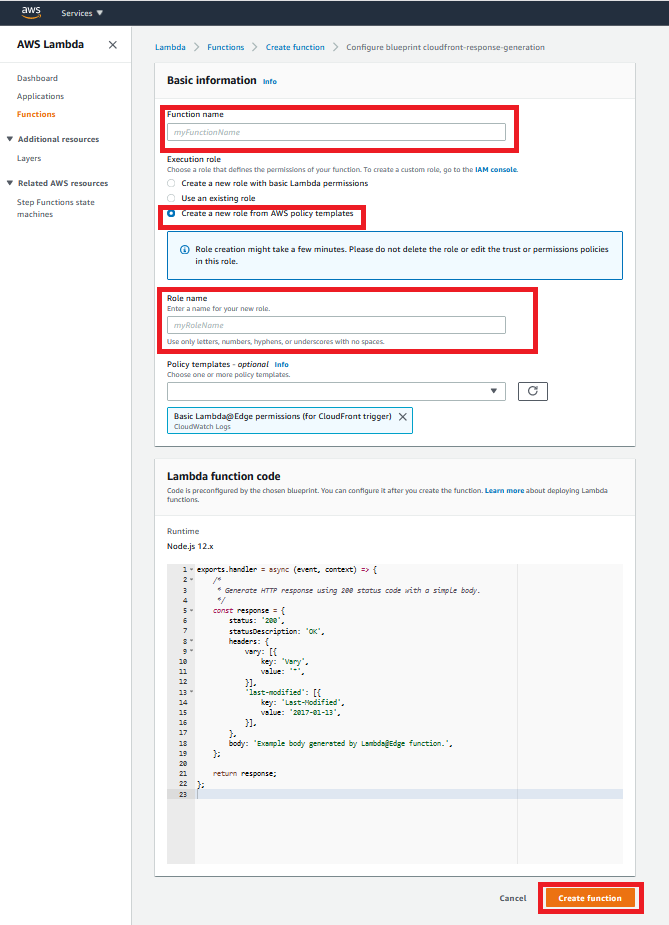
Click on **Create Function.**





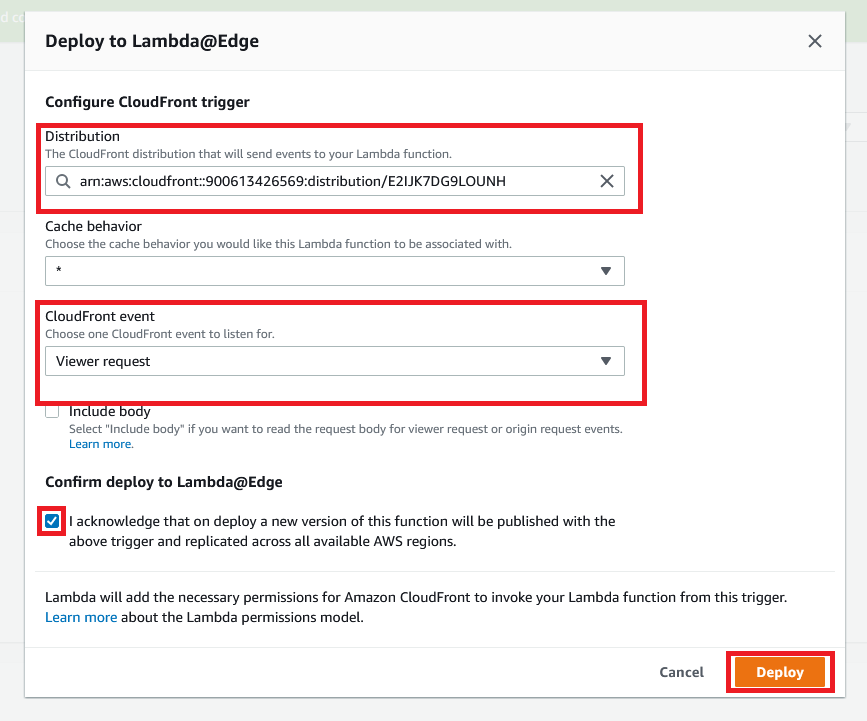
* Select the **Use a blueprint.**
* Search the **cloudfront-response-generation** then select it
* Then click on **configure** button.





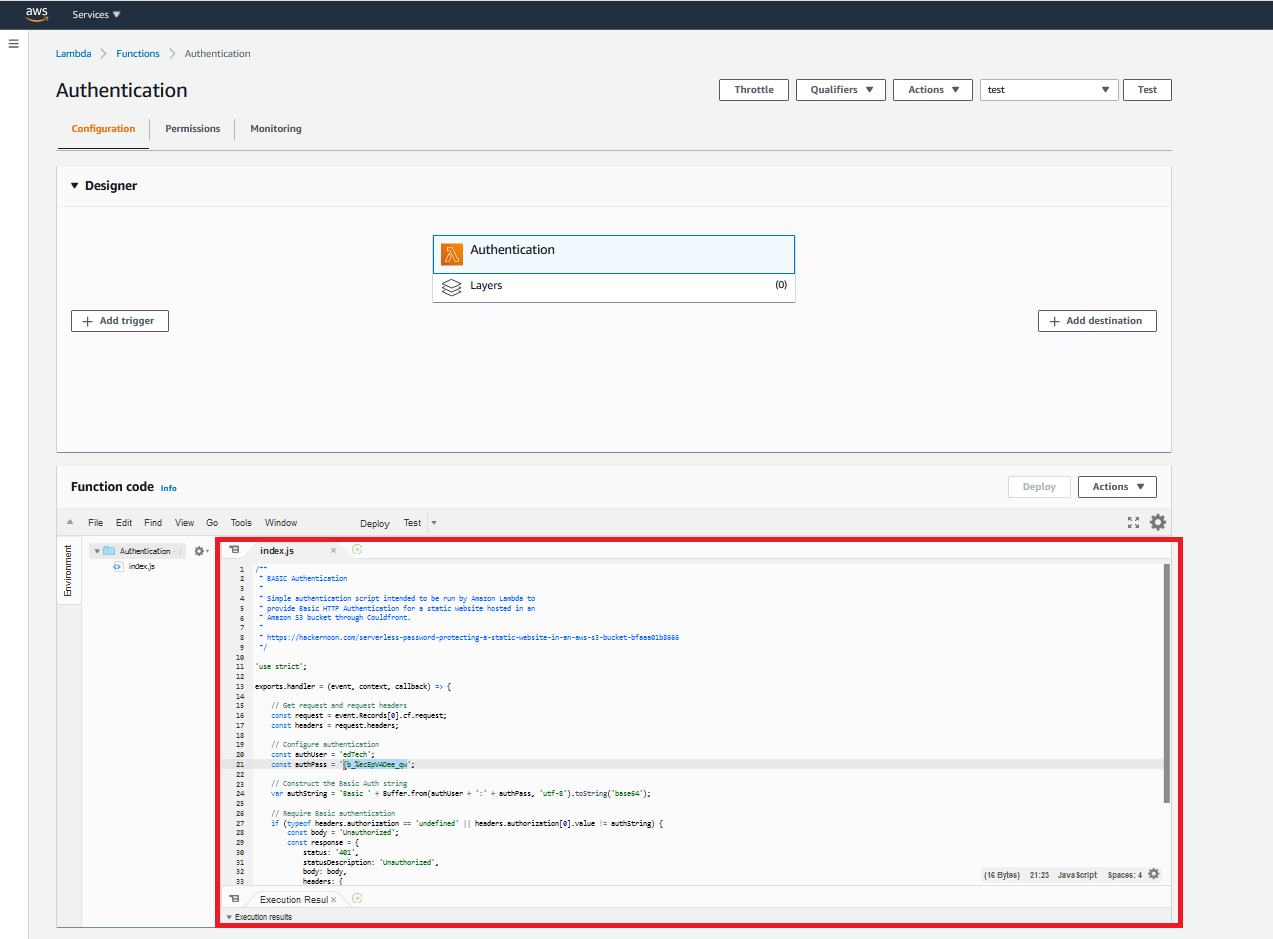
* Enter the Function Name
* Select the **Create a new role from AWS policy templates**
* Enter the Role Name.
* Click on **Create function** Button.





* Select the **Cloudfront Distribution ARN.**
* Select the CloudFront event: **Viewer request.**
* Select the **Confirm deploye to Lambda@Edge.**
* Then click on **Deploy** Button.





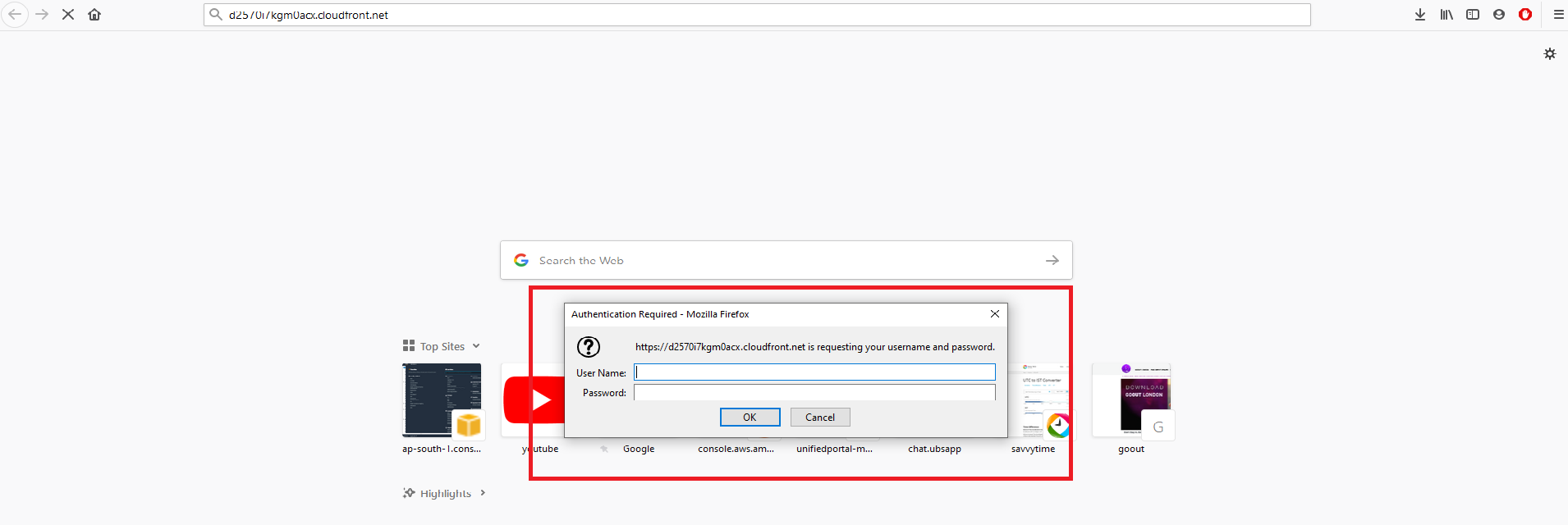
* Edit the **index.js** file and enter the authentication script.
* Then **Save** it.
* Then click the **Deploy** button.

**Script for Authentication:**

|  |
| --- |
| /\*\*  \* BASIC Authentication  \*  \* Simple authentication script intended to be run by Amazon Lambda to  \* provide Basic HTTP Authentication for a static website hosted in an  \* Amazon S3 bucket through Couldfront.  \*  \* https://hackernoon.com/serverless-password-protecting-a-static-website-in-an-aws-s3-bucket-bfaaa01b8666  \*/    'use strict';    exports.handler = (event, context, callback) => {    // Get request and request headers  const request = event.Records[0].cf.request;  const headers = request.headers;    // Configure authentication  const authUser = 'edTech';  const authPass = '{b\_%ecEpV4Dee\_qw';    // Construct the Basic Auth string  var authString = 'Basic ' + Buffer.from(authUser + ':' + authPass, 'utf-8').toString('base64');    // Require Basic authentication  if (typeof headers.authorization == 'undefined' || headers.authorization[0].value != authString) {  const body = 'Unauthorized';  const response = {  status: '401',  statusDescription: 'Unauthorized',  body: body,  headers: {  'www-authenticate': [{key: 'WWW-Authenticate', value:'Basic'}]  },  };  callback(null, response);  }    // Continue request processing if authentication passed  callback(null, request);  }; |

* Then **Save** it.
* Then click the **Deploy** button.





* Then search your website and look like this prompt the username and password authentication.