Project:

Password Protected Website

Use:
Password-protected websites on AWS provide a strong and flexible way to manage who can access content stored in the cloud.

Resources Requirment:

- -s3
- -cloudfront
- -lambda

+step1: Create S3 Bucket and upload website related file in it.

- Create s3 bucket with all default settings. (Note: checkbox-block all public access and ACLs enabled because bucket will be private only)
- Upload Website related in it

+Step2: Making Dynamic website using cloudfront.

- Go to Cloudfront and Create Distribution (we can use cloudfront for hosting the dynamic website)
- Origin Domain select existing created s3 bucket which have website uploaded in it.
- Origin Access select Legacy access identities (for accessing our recently created s3 bucket) create new OAI (origin access identity) already selected our bucket hit create then bucket policy will be yes (for read access of S3)
- Web Application Firewall (WAF) in that select do not enable security protection (for not need any protection)
- Then default root object is index.html (it just name of your object)
- Then hit create distribution
- Copy Distribution domain name paste into browser (for testing purpose whether our website is visible or not)

+Step3: Configuring Lambda function (or we can deploy code for protection)

- Go to lambda and Create Function
- Select author from scratch (because we can use our builded code) choose runtime node.js 16.x (because our code is in that format)
- Click on Change default execution role select Create a new role from AWS policy templates (for creating our custom role or we can create our own policy)
- Give name to the role.
- In policy templates we can select Basic Lambda@Edge permissions (for CloudFront trigger)
- Then Hit Create Function.

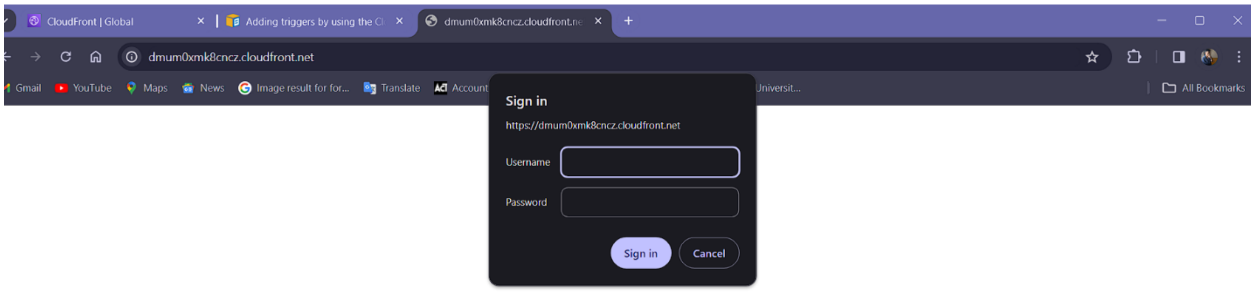
+Step4: Set username and password.

- Copy given source code from the developer (code is about username and password or protection purpose) and enter or edit our custom username password in that code.
- Go to file and save that code.
- Then hit Deploy.
- Then go to action and deploy to lambda@edge (now your code will applicable to the website)
- In distribution select existing created cloudfront distribution
- In cache behaviour select * which will get associated with lambda function
- CloudFront event select 'viewer request.' (it will only view not execute or not write)
- Check the include body and confirm deploy to Lambda@Edge
- Then hit Deploy.

+Step5: verification

- Go to Cloudfront
- Click on Distribution Id
- Copy the Distribution Domain Name and paste it into the browser

+Result:-



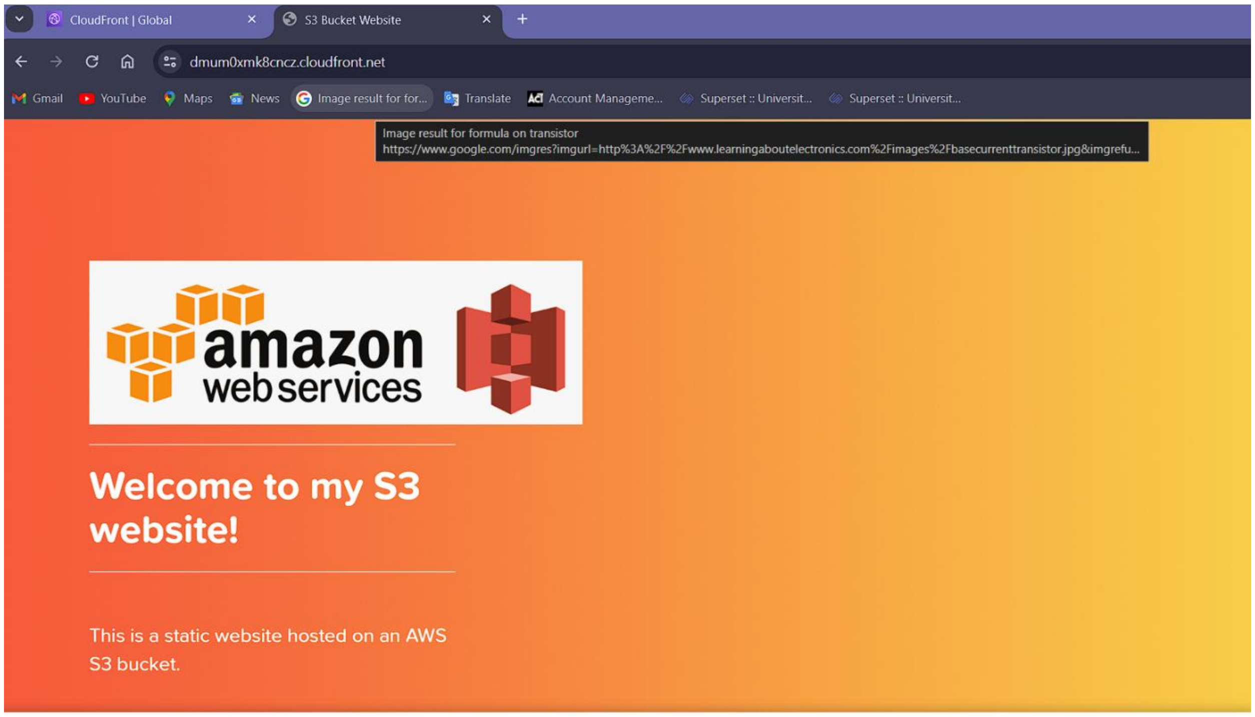
Sign in

https://dmum0xmk8cncz.cloudfront.net

Username

Password

❖ As we can see that website will be protected it will ask username and password.


CloudFront | Global
S3 Bucket Website
dmum0xmk8cncz.cloudfront.net
Gmail
YouTube
Maps
News
Image result for for...
Translate
Account Manageme...
Superset :: Universit...
Superset :: Universit...
Image result for formula on transistor
https://www.google.com/imgres?imgurl=http%3A%2F%2Fwww.learningaboutelectronics.com%2Fimages%2Fbasecurrenttransistor.jpg&imgrefu...
amazon web services

Welcome to my S3 website!

This is a static website hosted on an AWS S3 bucket.

❖ Step2 result website is visible or not

+Given source code for protection: -

```
'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 = 'user';
    const authPass = 'pass';

    // Construct the Basic Auth string
    const authString = 'Basic ' + new Buffer(authUser + ':' + authPass).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);
};
```

+Given Website Related File: -

https://drive.google.com/drive/folders/1bkpgjZnqpMqUC10dS3wUqCv4JdD41O5o?usp=drive_link