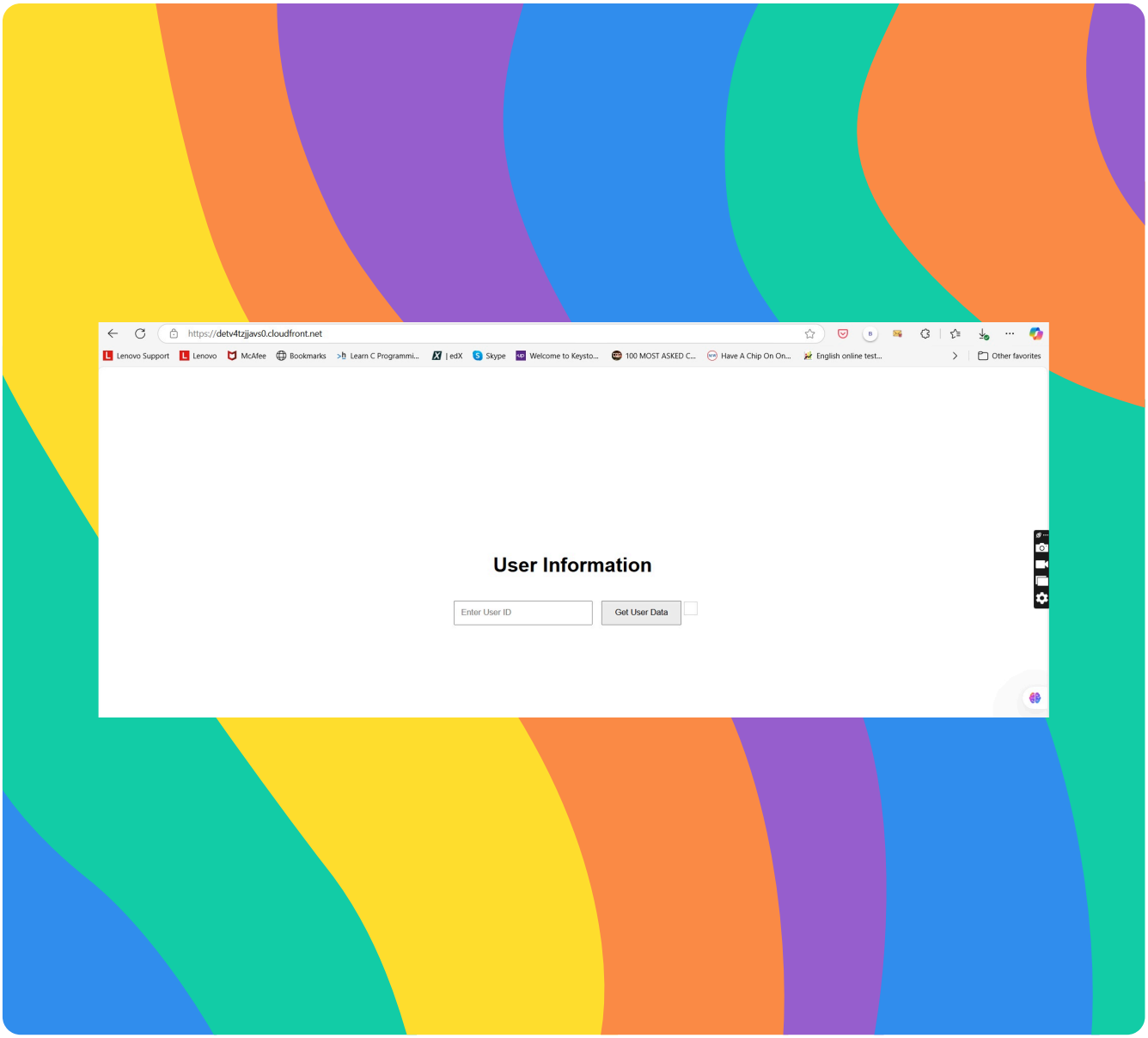
nextwork.org

Website Delivery with CloudFront

DI **Dineshraj Dhanapathy**



DI
Dineshraj Dhanapathy
NextWork Student

Introducing Today's Project!

In this project, I will demonstrate how to create a storage space in Amazon S3 for hosting website files, set up CloudFront to distribute the content globally with low latency, and manage permissions securely for both S3 and CloudFront. I'm doing this project to learn how to build and deploy a scalable, secure, and globally accessible website using AWS services. Additionally, I will compare different website hosting methods—such as S3 static hosting—analyzing their performance, cost, and use cases to gain deeper insights into choosing the right hosting solution for different scenarios.

Tools and concepts

Services I used were Amazon S3 for storing website files and enabling static website hosting, and Amazon CloudFront for distributing the website globally. Key concepts I learnt include content delivery network (CDN), which improves performance by caching content closer to users, origin access control (OAC) for securing S3 buckets, bucket policies to manage permissions, and the differences in performance, security, and configuration between using S3 directly versus through CloudFront.

DI
Dineshraj Dhanapathy
NextWork Student
NextWork.org

Project reflection

This project took me approximately 2 hours to complete. The most challenging part was configuring the correct permissions for both S3 and CloudFront to work together without triggering access denied errors. It was most rewarding to finally see my website live through a secure CloudFront distribution, knowing it was delivered efficiently and safely across the globe.

I did this project today to deepen my understanding of hosting and delivering websites using AWS services like S3 and CloudFront. I wanted hands-on experience with real-world cloud configurations, especially around permissions and content delivery. This project met my goals by helping me learn how to securely host a website, manage access, and optimize performance through a CDN. It also boosted my confidence in applying these skills to future cloud projects.

DI
Dineshraj Dhanapathy
NextWork Student

Set Up S3 and Website Files

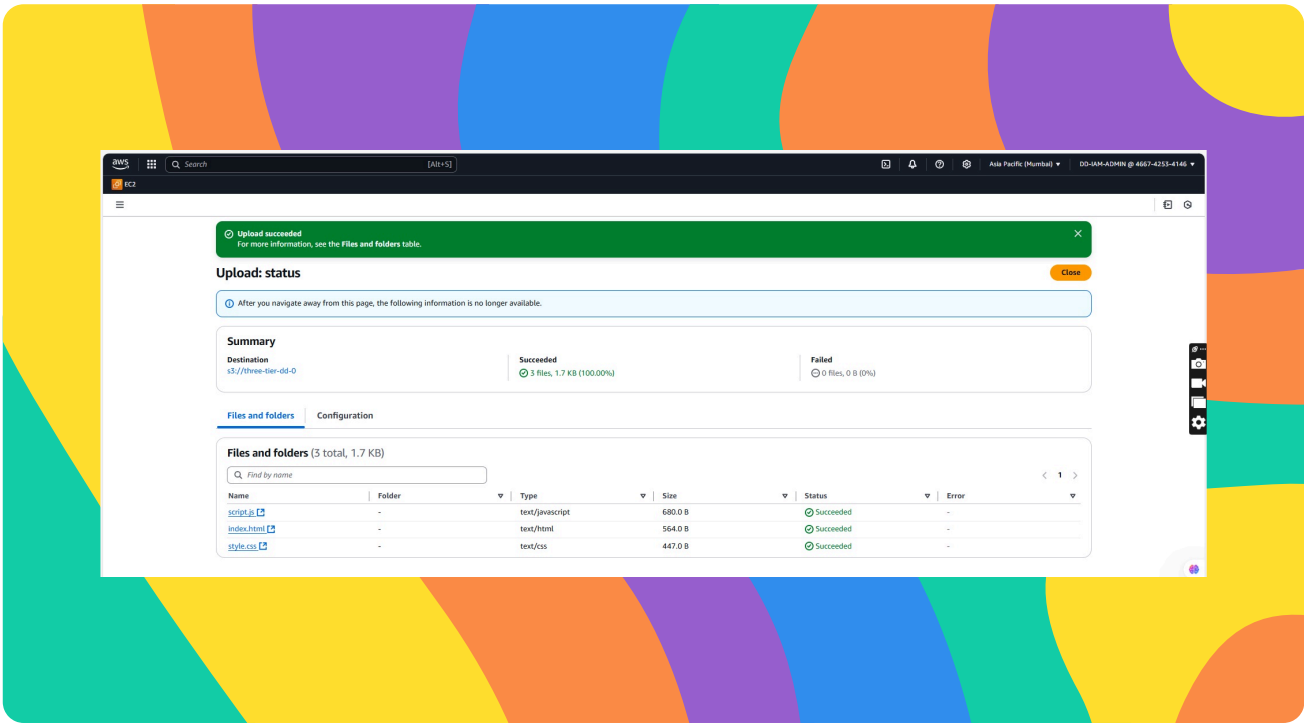
I started the project by creating an S3 bucket to store and manage all the files that make up my website—such as HTML, CSS, JavaScript, and images. S3 acts as a reliable and scalable storage solution where these assets can live securely in the cloud. I can't use CloudFront for this task because CloudFront is a content delivery network (CDN), not a storage service. It speeds up the delivery of content but depends on an origin source like S3 to fetch that content and distribute it globally.

The three files that make up my website are index.html, which contains the structure and content of the webpage; style.css, which defines the visual appearance and layout through styles and colors; and script.js, which adds interactivity and dynamic behavior to the website. Together, these files build a complete static website that is visually appealing and functionally responsive.

I validated that my website files work by opening the index.html file in a web browser to check if the page loads correctly, displays the intended layout, and functions as expected. I confirmed that the style.css is applying the correct styles and that the script.js is executing any interactive features properly. This ensured all files are linked correctly and the website behaves as intended before uploading to S3.

DI

Dineshraj Dhanapathy
NextWork Student



DI
Dineshraj Dhanapathy
NextWork Student

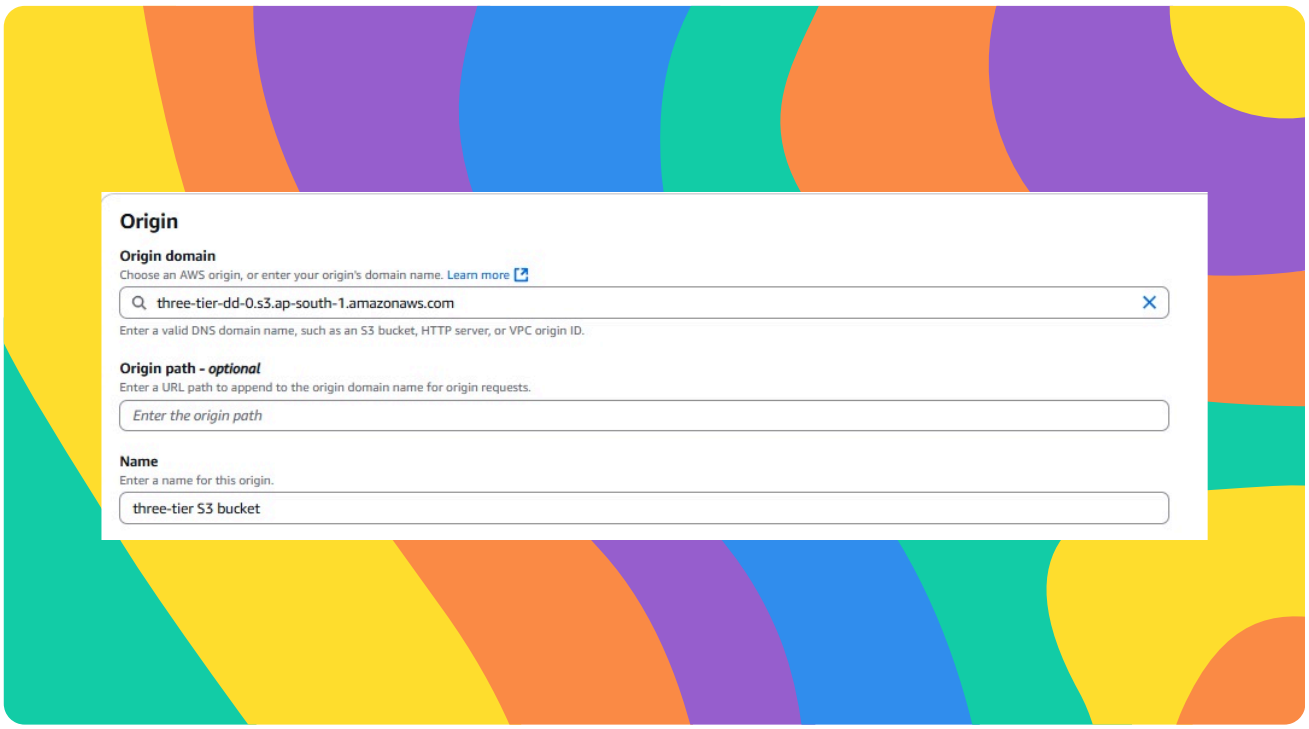
Exploring Amazon CloudFront

Amazon CloudFront is a content delivery network, which means it speeds up the delivery of static and dynamic web content—like HTML, CSS, JavaScript, and images—by caching it at edge locations around the world. This allows users to access content from a server that's physically closer to them, reducing latency and improving load times. Businesses and developers use CloudFront because it enhances website performance, ensures faster content delivery, improves user experience globally, and provides built-in security features like DDoS protection and HTTPS support.

To use Amazon CloudFront, you set up distributions, which are settings that define how CloudFront delivers your content and where it gets that content from. I set up a distribution for my website to ensure it loads quickly and efficiently for users across the globe. The origin is my S3 bucket, which stores all the files that make up my website, such as HTML, CSS, and JavaScript. This setup allows CloudFront to cache and serve content from the nearest edge location to the user.

DI
Dineshraj Dhanapathy
NextWork Student

My CloudFront distribution's default root object is index.html. This means when users visit my website without specifying a file name, CloudFront automatically serves the index.html file from the origin, which is my S3 bucket. I also configured the origin settings to point to my S3 bucket, enabled static website hosting, and set the viewer protocol policy to redirect HTTP to HTTPS for secure access. This setup ensures smooth loading and a secure experience for users.



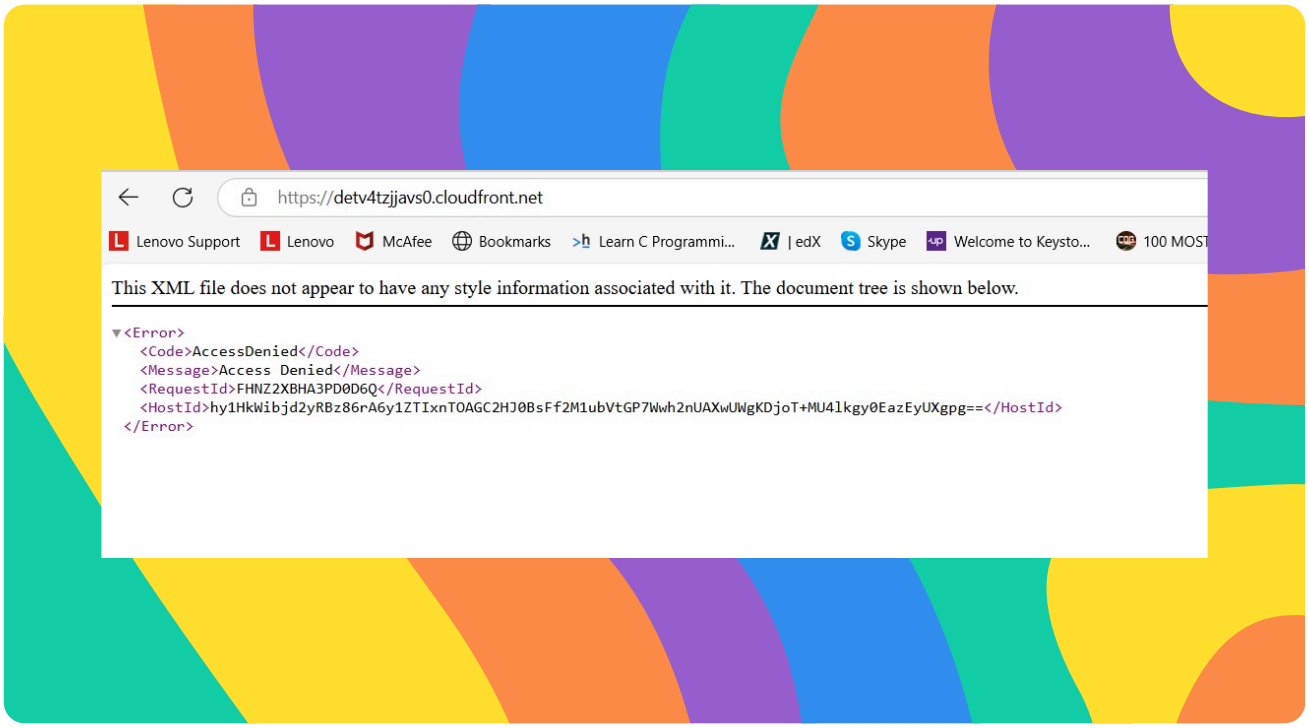
DI
Dineshraj Dhanapathy
NextWork Student
NextWork.org

Handling Access Issues

When I tried visiting my distributed website, I ran into an access denied error because CloudFront didn't have permission to access the files in my S3 bucket. By default, S3 buckets are private to protect data, and I hadn't yet configured the necessary permissions or set up Origin Access Control (OAC) to allow CloudFront to retrieve and serve the website content. Without these permissions, CloudFront is blocked from accessing the bucket, resulting in the error.

My distribution's origin access settings were set to public, meaning the S3 bucket was accessible in theory. This caused the access denied error because the individual objects in the S3 bucket—like index.html, style.css, and script.js—were still private by default. Even with public origin access, CloudFront couldn't retrieve the content until I updated the object permissions to allow public access, ensuring that all users, including CloudFront, could read the files.

To resolve the error, I set up origin access control (OAC). OAC is a secure way to give CloudFront permission to access content in my S3 bucket without making the bucket or its objects publicly accessible. It acts like a special CloudFront user that can fetch files from S3 while keeping them private to everyone else. OAC helps protect against unauthorized access and gives me fine-grained control over how and when CloudFront can retrieve my content. This way, only users accessing the site through CloudFront can see the content, improving both security and performance.

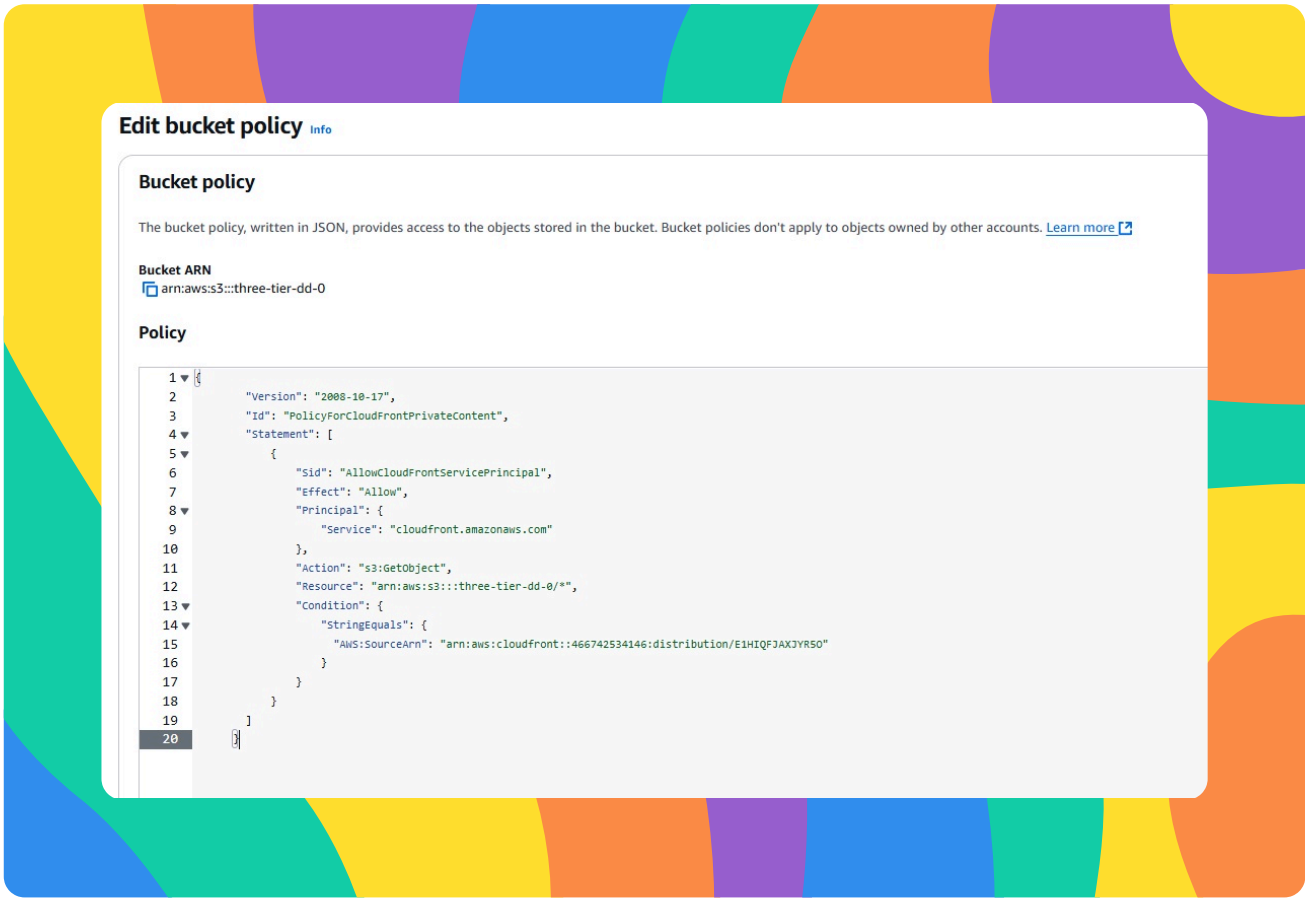


DI
Dineshraj Dhanapathy
NextWork Student

Updating S3 Permissions

Once I set up my OAC, I still needed to update my bucket policy because the bucket must explicitly allow access from the OAC's identity. Without this update, S3 will still block CloudFront's requests, even though the OAC exists. The bucket policy acts as a gatekeeper and needs to include permissions that let the OAC access the necessary objects, ensuring CloudFront can serve the content securely and correctly.

Creating an OAC automatically gives me a policy I could copy, which grants CloudFront permission to access the objects in my S3 bucket securely. This policy allows only the specific OAC identity to perform actions like s3:GetObject, ensuring that no one else can access the files directly. It keeps my bucket private while enabling CloudFront to serve content to users, combining both security and performance in the delivery of my website.


Edit bucket policy Info
Bucket policy
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more
Bucket ARN
arn:aws:s3:::three-tier-dd-0
Policy
{
"Version": "2008-10-17",
"Id": "PolicyForCloudFrontPrivateContent",
"Statement": [
{
"Sid": "AllowCloudFrontServicePrincipal",
"Effect": "Allow",
"Principal": {
"Service": "cloudfront.amazonaws.com"
},
"Action": "s3:GetObject",
"Resource": "arn:aws:s3:::three-tier-dd-0/*",
"Condition": {
"StringEquals": {
"AWS:SourceArn": "arn:aws:cloudfront::466742534146:distribution/E1HIQFJAXJYR5O"
}
}
}
]
}

DI
Dineshraj Dhanapathy
NextWork Student
NextWork.org

S3 vs CloudFront for Hosting

For my project extension, I'm comparing S3 static website hosting with CloudFront. I initially had an error with static website hosting because I hadn't enabled the "Static website hosting" option in the S3 bucket settings or set the correct index and error documents. Additionally, my bucket policy might not have allowed public access to the objects, which is required for direct access via the S3 website endpoint.

I tried resolving this by unchecking "Block all public access" in my S3 bucket settings, but I still ran into an error because I didn't add a bucket policy to explicitly grant public read permissions. Disabling the block just removes the restriction, but it doesn't actually allow access on its own. Without a proper bucket policy that tells AWS to allow public users to read the objects, the files remain inaccessible, leading to continued access denied errors.

I could finally see my S3 hosted website when I updated the bucket policy to allow public read access to all objects and made sure each file was set to public. This worked because the policy explicitly granted permission for anyone to access the content, and with static website hosting enabled, S3 could now serve the files through its website endpoint without any access restrictions.

Compared to the permission settings for my CloudFront distribution, using S3 meant I had to make my bucket and objects publicly accessible, which raised potential security concerns. CloudFront, on the other hand, allowed me to keep my bucket private by using Origin Access Control (OAC), granting access only through CloudFront. I preferred CloudFront's approach because it provided more control and security while still delivering content efficiently.

DI
Dineshraj Dhanapathy
NextWork Student

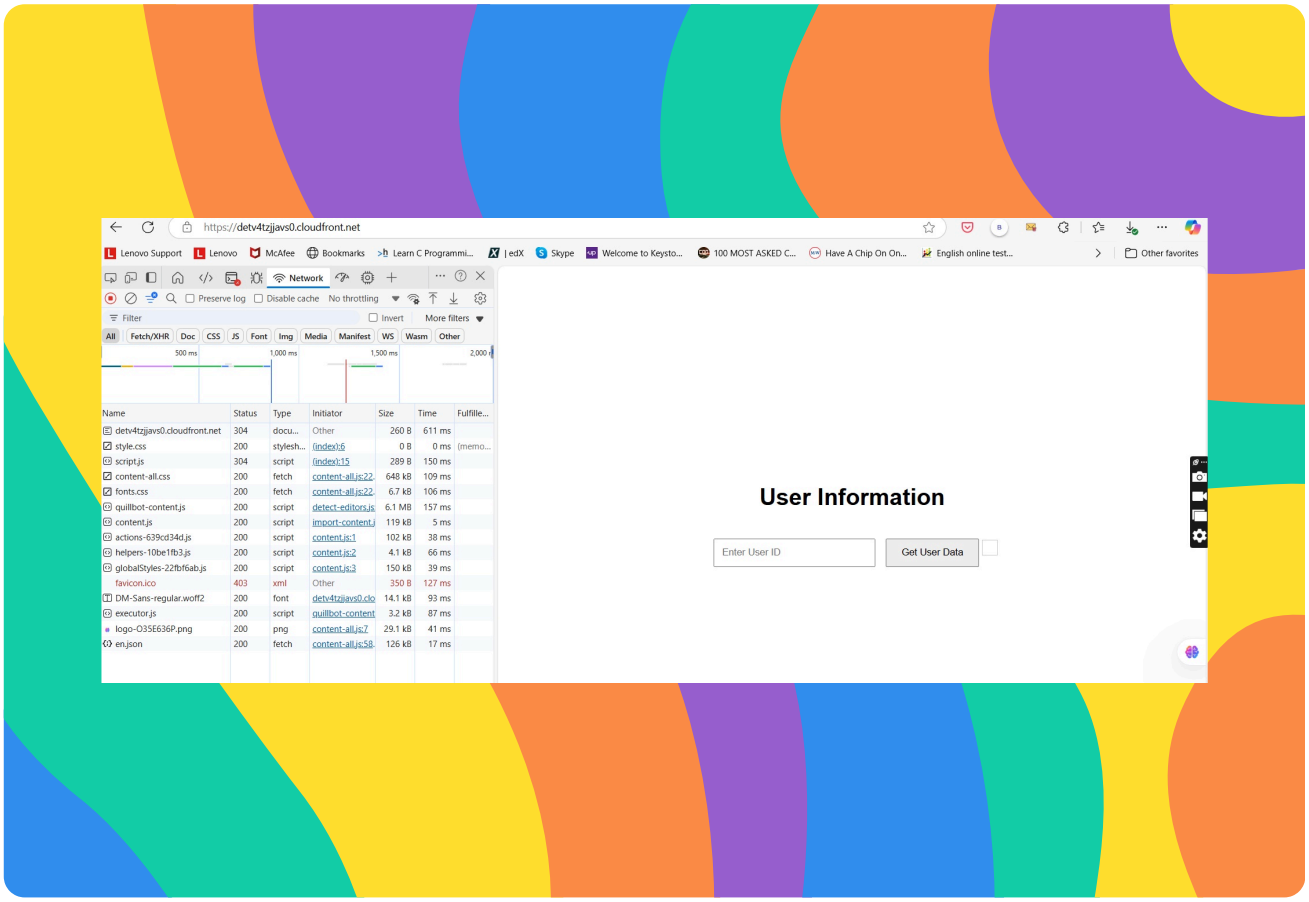
S3 vs CloudFront Load Times

Load time means the amount of time it takes for a website's content to appear on the user's screen after they request it. The load times for the CloudFront site were faster than the S3 site because CloudFront uses a global network of edge locations to cache and deliver content closer to the user. In contrast, S3 static website hosting serves files from a single AWS region, causing longer travel times for users far from that region, which slows down performance.

A business would prefer CloudFront when they need fast, secure, and reliable global delivery of website content, especially for users spread across different regions. CloudFront offers caching, HTTPS, and better control over access. S3 static website hosting might be sufficient when the site is simple, has low traffic, or serves a local audience where speed and advanced features are less critical. It's a good low-cost option for basic hosting needs.

DI

Dineshraj Dhanapathy
NextWork Student



Everyone should be in a job they love.

Check out nextwork.org for more projects