

# Step 1 - How to deploy Frontends to AWS



- New things we will learn include
- 1. Object stores (S3)
- 2. CDNs (Cloudfront)

Step 1 - Signup and get an AWS account.

Step 2 - Make sure you can access S3 and cloudfront (this will automatically happen if you are the root user of that account)

The screenshot shows the AWS Console Home page. On the left, under 'Recently visited' services, 'S3' and 'CloudFront' are highlighted with red arrows pointing towards the right side of the screen. The right side displays the 'Applications' section, which is currently empty. The 'Create application' button is visible at the top of this section.

# Step 2 - Build your React frontend



This approach will not work for frameworks that use Server side rendering (like Next.js)  
This will work for basic React apps, HTML/CSS/JS apps

## Go to your react project

```
cd /link/to/your/react/project
```

Copy

## Build your project

```
npm run build
```

Copy

## Try serving the HTML/CSS/JS locally

```
npm i -g serve  
serve
```

Copy

At this point you have basic HTML/CSS/JS code that you can deploy on the internet.

You might be tempted to host this on an EC2 instance, but that is not the right approach

The next slide explains why

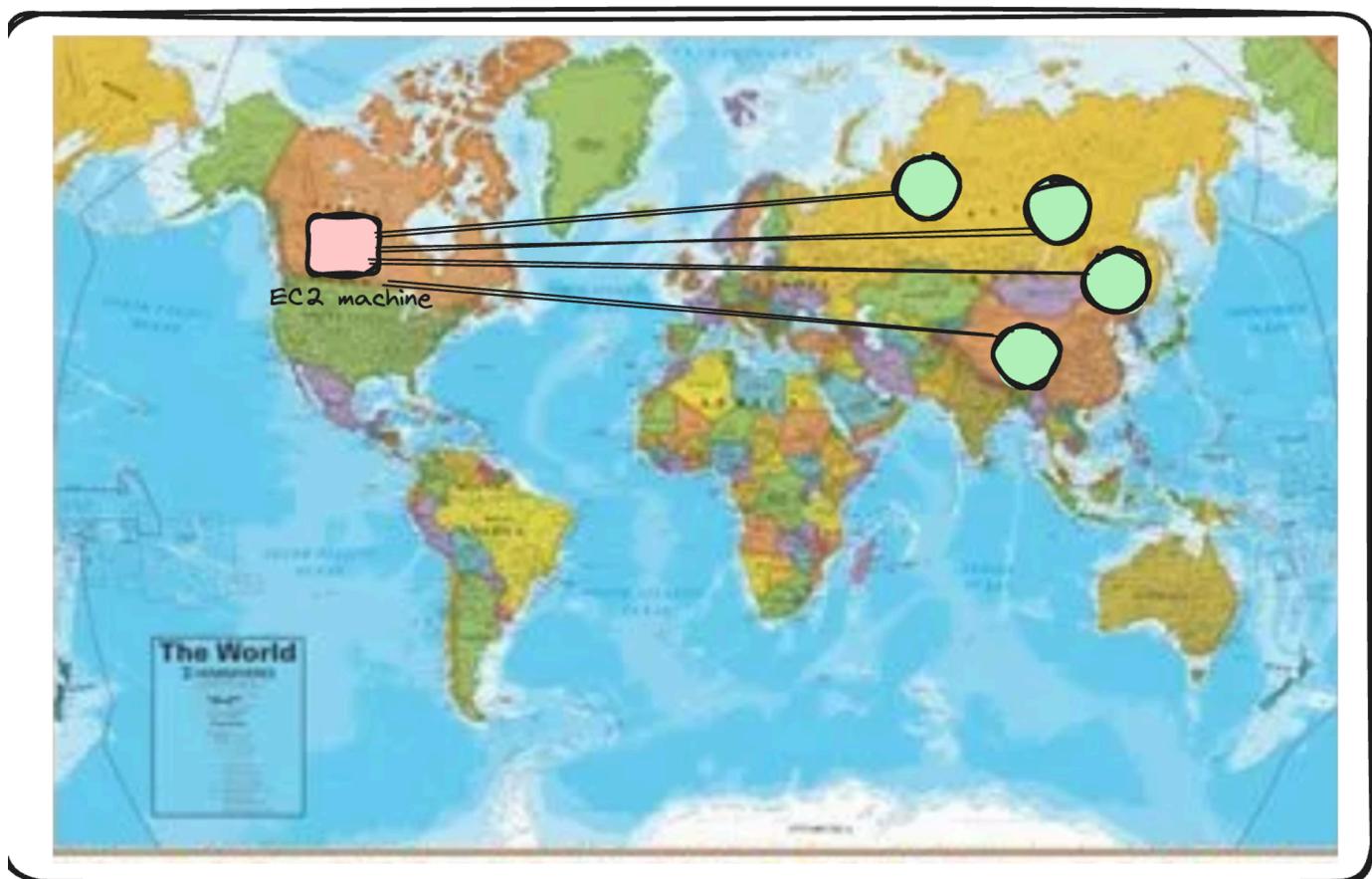
# Step 3 - What are CDNs?

A CDN stands for **Content Delivery Network**.

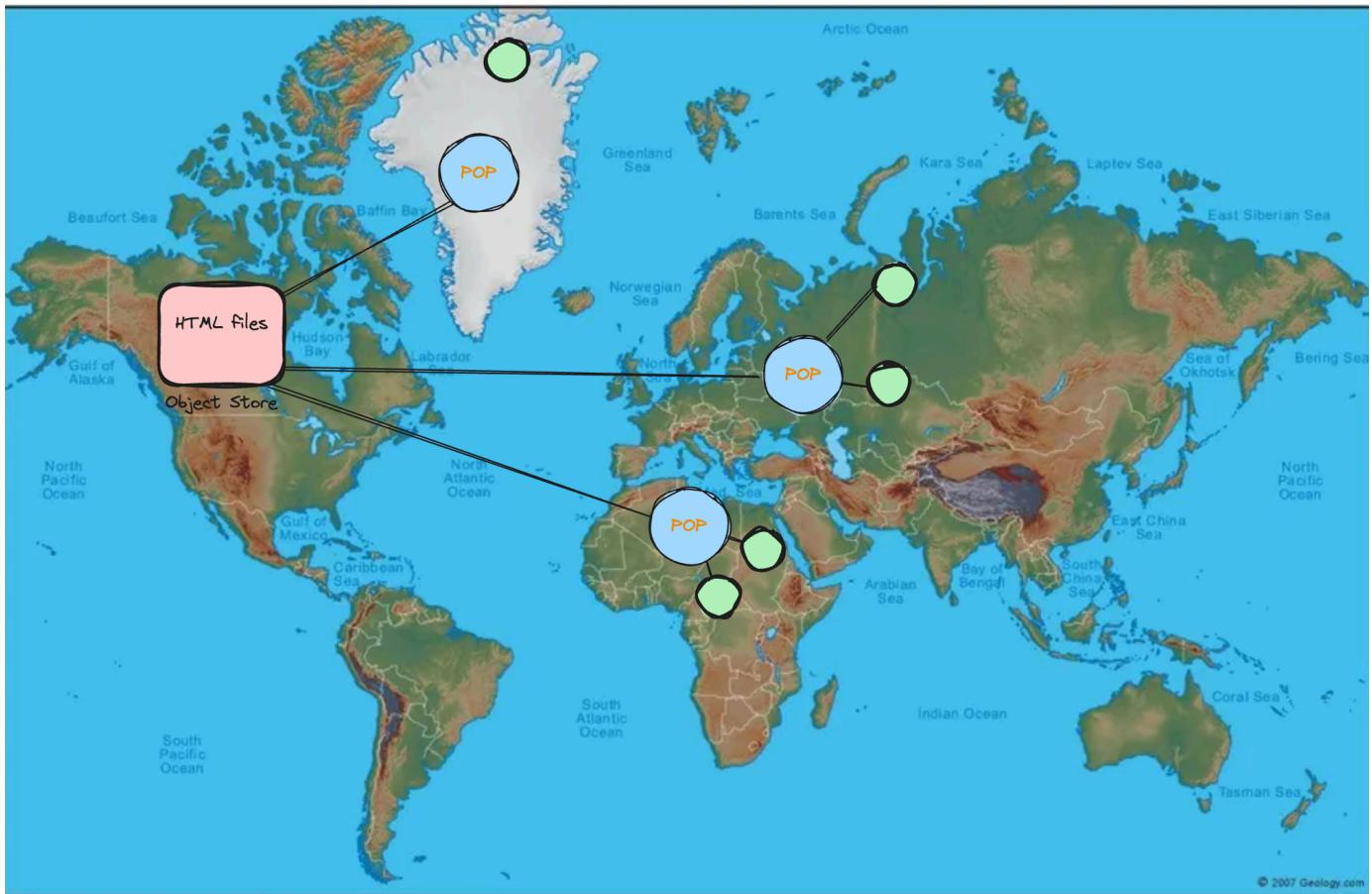
As the name suggests, it's an optimal way for you to deliver content (mp4 files, jpgs and even HTML/CSS/JS files) to your users.

It is better than serving it from a VM/EC2 instances because of a few reasons -

## 1. EC2 machine approach



## 2. CDN approach



1. For frontends, mp4 files, images, **Object stores + CDNs** are a better approach.
2. You can't use the same for backends, since every request returns a different response.  
Caching doesn't make any sense there.



You can use edge networks for backends (deploy your backend on various servers on the internet) but data can't be cached in there.

Great video on how Hotstar scales their infrastructure during cricket matches (they use CDNs heavily)

**How Hotstar Scaled 25 Million Users**

Concurrency Pattern!

Gaurav Kamboj  
Cloud Architect at Hotstar  
@eyehoeye

100 K Views

hotstartech\_

## Step 4 - Creating an object store in AWS

In AWS, S3 is their object store offering.

You can create a **bucket** in there. A **bucket** represents a logical place where you store all the files of a certain project.

Name	AWS Region	Access	Creation date
test11123123	Asia Pacific (Mumbai) ap-south-1	Bucket and objects not public	June 10, 2023, 21:51:06 (UTC+02:00)

[Amazon S3](#) > [Buckets](#) > [Create bucket](#)

## Create bucket Info

Buckets are containers for data stored in S3. [Learn more](#)

### General configuration

#### AWS Region

Asia Pacific (Mumbai) ap-south-1

#### Bucket name Info

kirat-test

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

#### Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Format: s3://bucket/prefix

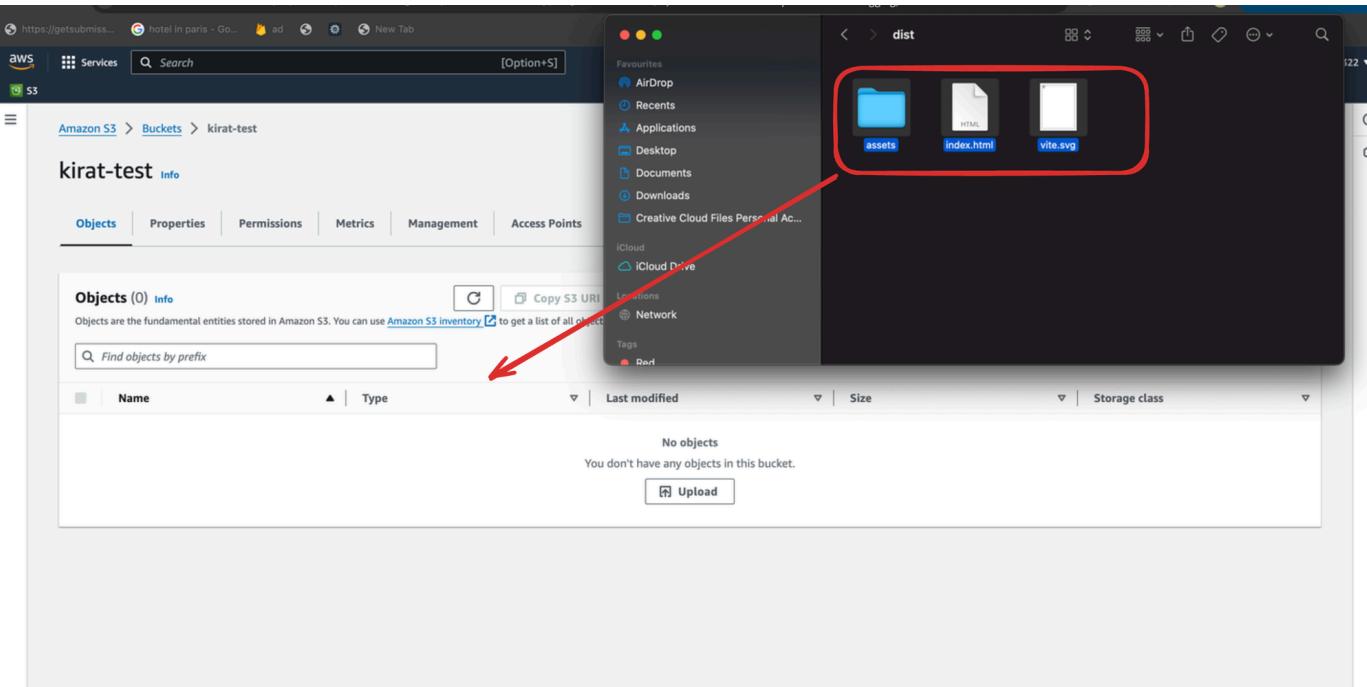
### Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership



# Step 5 - Upload the file bundle to S3

Upload all the files in the `dist` folder of your react project to S3



## Step 6 - Try accessing the website

This screenshot shows a browser window with the URL [kirat-test-2.s3.eu-west-3.amazonaws.com/index.html](https://kirat-test-2.s3.eu-west-3.amazonaws.com/index.html). The page content is an XML error document:

```
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestId>T3YE0RGZZXH90J2W</RequestId>
<HostId>HGnifF+pOhpxbx3fl199MwqAQN5Y5bVVN7hmLexMojiy9bk1wUCKqXDwUm3tuXCqI9fkli3jMQ=</HostId>
</Error>
```

You might be tempted to open your S3 bucket at this point, but don't

Your S3 bucket should be blocked by default, and you should allow CloudFront (CDN) to access it.

# Step 7 - Connecting Cloudfront

## Step 1 - Create Cloudfront distribution

Go to CloudFront and create a new distribution. A **distribution** here means you're creating a place from where **content** can be distributed.

Distributions (11) [Info](#)

ID	Description	Type	Domain name	Alternate do...	Origins	Status	Last modified
<a href="#">E3JCIJ900E2RYZ</a>	-	Production	dibs5cabw92oe...	fe.100xdevs.com	kirat-test-2.s3.eu-wes	<span>Enabled</span>	February 19, 20...
<a href="#">E2B3PG65NKSQMO</a>	-	Production	d2bq1fmgpm8...	-	kirat-test.s3-website.i	<span>Enabled</span>	February 19, 20...

[Create distribution](#)

## Step 2 - Select your S3 bucket as the source

### Origin

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

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

**Name**  
Enter a name for this origin.

**Origin access** | [Info](#)

- Public  
Bucket must allow public access.
- Origin access control settings (recommended)  
Bucket can restrict access to only CloudFront.
- Legacy access identities  
Use a CloudFront origin access identity (OAI) to access the S3 bucket.

**Origin access control**  
Select an existing origin access control (recommended) or create a new control.

Create new OAC

⚠ This field cannot be empty

⚠ **You must update the S3 bucket policy**  
CloudFront will provide you with the policy statement after creating the distribution.



Origin Access Control (OAC) is a feature in Cloudfront, which allows you to restrict direct access to the content stored in your origin, such as an Amazon S3 bucket or a

web server, ensuring that users can only access the content through the CDN distribution and not by directly accessing the origin URL.

By the end of this, you should have a working cloudfront URL.

[CloudFront](#) > [Distributions](#) > E3JCIJ9O0E2RYZ

### E3JCIJ9O0E2RYZ

[View metrics](#)

[General](#) | [Security](#) | [Origins](#) | [Behaviors](#) | [Error pages](#) | [Invalidations](#) | [Tags](#)

**Details**

Distribution domain name <a href="#">dibs5cabw92oe.cloudfront.net</a>	ARN <a href="#">arn:aws:cloudfront::163679972322:distribution/E3JCIJ9O0E2RYZ</a>	Last modified February 19, 2024 at 6:33:02 AM UTC
--	---	--

**Settings**

Description -	Alternate domain names fe.100xdevs.com Custom SSL certificate <a href="#">fe.100xdevs.com</a>	Standard logging Off Cookie logging Off Default root object
Price class <a href="#">Use all edge locations (best performance)</a>	Supported HTTP versions <a href="#">HTTP/2, HTTP/1.1, HTTP/1.0</a>	

# Step 8 - Connect your own domain to it

Websites aren't fun if you have to go to a URL that looks like this -  
<https://dibs5cabw92oe.cloudfront.net>

Connect your own custom domain by following the given steps -

## 1. Select edit on the root page

[CloudFront](#) > [Distributions](#) > E2B3PG65NKSQMO

### E2B3PG65NKSQMO

[View metrics](#)

[General](#) | [Security](#) | [Origins](#) | [Behaviors](#) | [Error pages](#) | [Invalidations](#) | [Tags](#)

**Details**

Distribution domain name <a href="#">d2bq1lmgpm8dp.cloudfront.net</a>	ARN <a href="#">arn:aws:cloudfront::163679972322:distribution/E2B3PG65NKSQMO</a>	Last modified February 19, 2024 at 5:01:19 AM UTC
--	---	--

**Settings**

Description -	Alternate domain names -	Standard logging Off Cookie logging Off Default root object -
Price class <a href="#">Use all edge locations (best performance)</a>	Supported HTTP versions <a href="#">HTTP/2, HTTP/1.1, HTTP/1.0</a>	

## 2. Attach a domain name to the distribution

**Settings**

**Price class** | [Info](#)  
Choose the price class associated with the maximum price that you want to pay.

Use all edge locations (best performance)  
 Use only North America and Europe  
 Use North America, Europe, Asia, Middle East, and Africa

**Alternate domain name (CNAME) - optional**  
Add the custom domain names that you use in URLs for the files served by this distribution.

fe.100xdevs.com [Remove](#)

[Add item](#)

ⓘ To add a list of alternative domain names, use the [bulk editor](#).

**Custom SSL certificate - optional**  
Associate a certificate from AWS Certificate Manager. The certificate must be in the US East (N. Virginia) Region (us-east-1).

[Choose certificate](#) ▾ [C](#)

[Request certificate](#)

**Supported HTTP versions**  
Add support for additional HTTP versions. HTTP/1.0 and HTTP/1.1 are supported by default.

HTTP/2  
 HTTP/3

**Default root object - optional**  
The object (file name) to return when a viewer requests the root URL (/) instead of a specific object.

## 3. Create a certificate

Since we want our website to be hosted on HTTPS, we should request a certificate for our domain

# Edit settings

## Settings

### Price class [Info](#)

Choose the price class associated with the maximum price that you want to pay.

- Use all edge locations (best performance)
- Use only North America and Europe
- Use North America, Europe, Asia, Middle East, and Africa

### Alternate domain name (CNAME) - *optional*

Add the custom domain names that you use in URLs for the files served by this distribution.

[Remove](#)
[Add item](#)

 To add a list of alternative domain names, use the [bulk editor](#).

### Custom SSL certificate - *optional*

Associate a certificate from AWS Certificate Manager. The certificate must be in the US East (N. Virginia) Region (us-east-1).


[Request certificate](#)

### Supported HTTP versions

## Step 4 - Follow steps to create the certificate in the certificate manager

Certificate status							
Identifier fb2088a7-c75f-4a42-91af-2a7fb4600a0e	Status  Issued	ARN <a href="#">arn:aws:acm:us-east-1:163679972322:certificate/fb2088a7-c75f-4a42-91af-2a7fb4600a0e</a>	Type Amazon Issued				
<b>Domains (1)</b>							
<a href="#">Create records in Route 53</a> <a href="#">Export to CSV</a>							
Domain      Status      Renewal status      Type      CNAME name      CNAME value							
fe.100xdevs.com	 Success	-	CNAME	<a href="#">_a96f3ef5c0e0ef282152985dfb428092.fe.100xdevs.com.</a>	<a href="#">_21845a5bfaa0b0cbbb6b8a55b28c5501.mhbtspdnt.acm-validations.aws.</a>		
Details							
In use	Serial number	Requested at	Renewal eligibility				

 These DNS settings are active. Changes are published immediately, but may take time to propagate

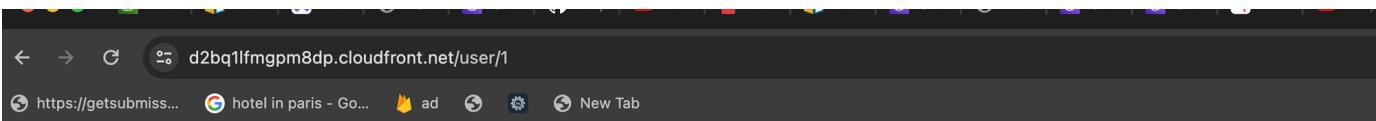
Resource records				 Export DNS records
Resource records point to the services that your domain uses, including web and email services. <a href="#">Learn more about resource records</a>				
<b>Custom records</b>				
100xdevs.com/A, _a96f3ef5c0e0ef282152985dfb428092.fe.100xdevs.com/CNAME and 27 more				
<a href="#">Manage custom records</a>				
Host name	Type	TTL	Data	
100xdevs.com	A	1 hour	76.76.21.21	
_a96f3ef5c0e0ef282152985dfb428092.f e.100xdevs.com	CNAME	1 hour	_21845a5bfaa0b0cbbb6b8a55b28c5501.mhbtsbpndt.acm-validations.aws.	

## Step 5 - Add a CNAME record for the website to point to your cloudfront URL

fe.100xdevs.com	CNAME	1 hour	dibs5cabw92oe.cloudfront.net.
-----------------	-------	--------	-------------------------------

## Step 9 - Error pages

You will notice a problem, whenever you try to access a route on your page that isn't the index route (/user/1) , you reach an error page



### 403 Forbidden

- Code: AccessDenied
- Message: Access Denied
- RequestId: V17AB7NEC9FRRDWX
- HostId: OJzE4K3MrighV9+NivXXYCblueDb26lEZ6MEVL99vUfhQZkiYXW9K1IUjtAvpFMRyx/IoMVnqaw=

This is because CloudFront is looking for a file `/user/1` in your S3, which doesn't exist.

To make sure that all requests reach `index.html`, add an `error page` that points to `index.html`

HTTP error code	Minimum TTL (seconds)	Response page path	HTTP response code
404: Not Found	0	/index.html	200: OK

## Edit error page response

**Error response Info**

**HTTP error code**  
Customize the custom error response when the origin sends this error code.

404: Not Found ▾

**Error caching minimum TTL**  
Enter the error caching minimum time to live (TTL), in seconds.

0

**Customize error response**  
Send a custom error response instead of the error received from the origin.

No

Yes

**Response page path**  
Enter the path to the custom error response page.

/index.html

**HTTP Response code**  
Choose the HTTP status code to return to the viewer. CloudFront can return a different status code to the viewer than what it received from the origin.

200: OK ▾

**Cancel** **Save changes**



You might have to invalidate cache to see this in action.

