AWS CloudFront

Introduction to Content Delivery Networks (CDN)

Imagine you have a website with lots of cool content, like images, videos, and documents. When a user visits your site from a different location far away from your server, the content might take a long time to load. That's where CDN comes to the rescue!

A CDN is like a network of servers spread across various locations worldwide. These servers store a copy of your website's content. When a user requests your website, the content is delivered from the server closest to the user, making it super fast! It's like having a local store for your website content everywhere in the world.

CloudFront

CloudFront is Amazon Web Services' (AWS) very own CDN service. It integrates seamlessly with other AWS services and allows you to deliver content, videos, applications, and APIs securely with low-latency and high transfer speeds.

How Does CloudFront Work

Let's understand how CloudFront works with a simple example:

Imagine you have a website with images stored on an Amazon S3 bucket (a cloud storage service). When a user requests an image, the request goes to CloudFront first.

Here's how the process flows:

- **Step 1:** CloudFront checks if it already has the requested image in its cache (storage). If it does, great! It sends the image directly to the user. If not, it proceeds to Step 2.

- **Step 2:** CloudFront fetches the image from the S3 bucket and stores a copy in its cache for future requests. Then, it sends the image to the user.

The next time someone requests the same image, CloudFront will deliver it from its cache, making it super fast and efficient!

4. Benefits of CloudFront

- **Fast Content Delivery:** CloudFront ensures your content reaches users with minimal delay, making your website lightning fast.
- **Global Reach:** With servers in various locations worldwide, CloudFront brings your content closer to users, regardless of where they are.
- **Security**: CloudFront provides security features like DDoS protection and SSL/TLS encryption to keep your content and users safe.
- **Scalability:** CloudFront can handle traffic spikes effortlessly, ensuring a smooth experience for your users.
- **Cost-Effective:** Pay only for the data transfer and requests made, making it cost-effective for businesses of all sizes.

5. Setting Up CloudFront on AWS

Now, let's get our hands dirty and set up CloudFront on AWS!

Step 1: Create an S3 Bucket

1. Go to the AWS Management Console and navigate to Amazon S3.
2. Create a new bucket to store your website content.

Step 2: Upload Content to the S3 Bucket

1. Upload images, videos, or any other content you want to serve through CloudFront to your S3 bucket.

Step 3: Create a CloudFront Distribution

1. Go to the AWS Management Console and navigate to CloudFront.
2. Click "Create Distribution."
3. Choose whether you want to deliver a web application or content (like images and videos).
4. Configure your settings, such as the origin (your S3 bucket), cache behaviors, and security settings.
5. Click "Create Distribution" to set up CloudFront.

Step 4: Update Website URLs

1. Once your CloudFront distribution is deployed (it may take a few minutes), you'll get a CloudFront domain name (e.g., d1a2b3c4def.cloudfront.net).
2. Replace the URLs of your website content with the CloudFront domain name.

That's it! Your content is now being delivered through CloudFront.

6. Use Cases and Scenarios

Scenario 1: E-Commerce Website

Let's say you have an e-commerce website that sells products globally. By using CloudFront, your product images and videos load quickly for customers all over the world, improving the shopping experience.

Scenario 2: Media Streaming

You're running a video streaming platform. With CloudFront, you can stream videos to users efficiently, regardless of their location, without buffering issues.

Scenario 3: Software Downloads

If you offer software downloads, CloudFront can distribute your files faster, reducing download times and providing a better user experience.

7. Tips and Best Practices

- Caching Strategies: Configure cache settings wisely to balance freshness and speed for different types of content.
- Invalidation: Learn how to invalidate or clear cached content when you make updates to your website.
- Monitoring and Reporting: Use AWS tools to monitor your CloudFront distribution's performance and gain insights into user behavior.

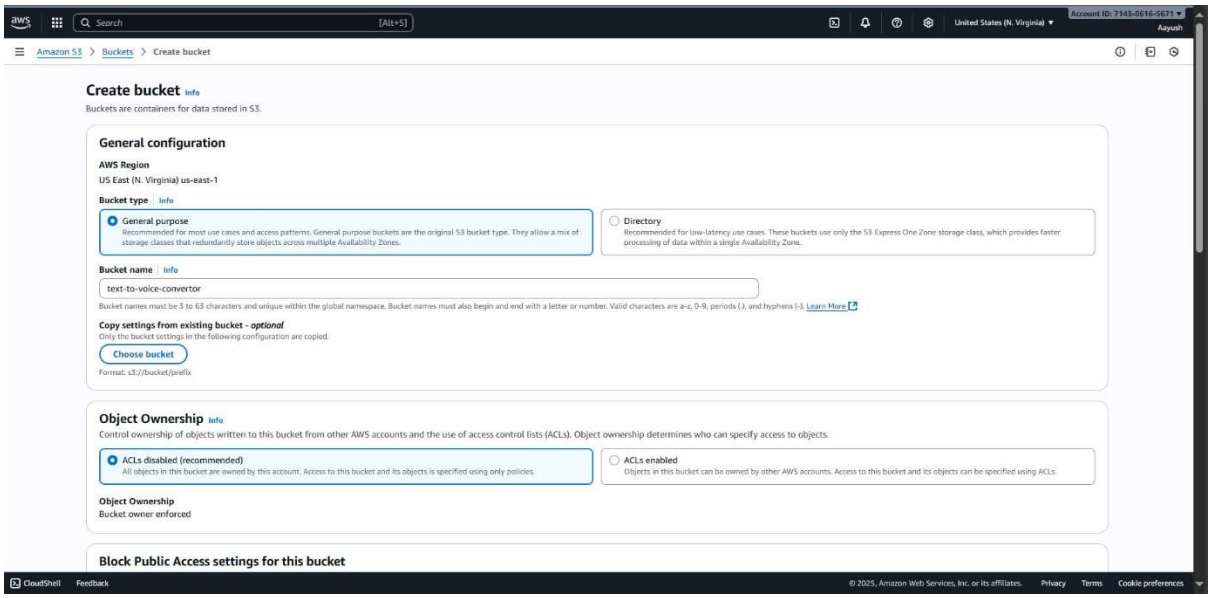
8. Conclusion

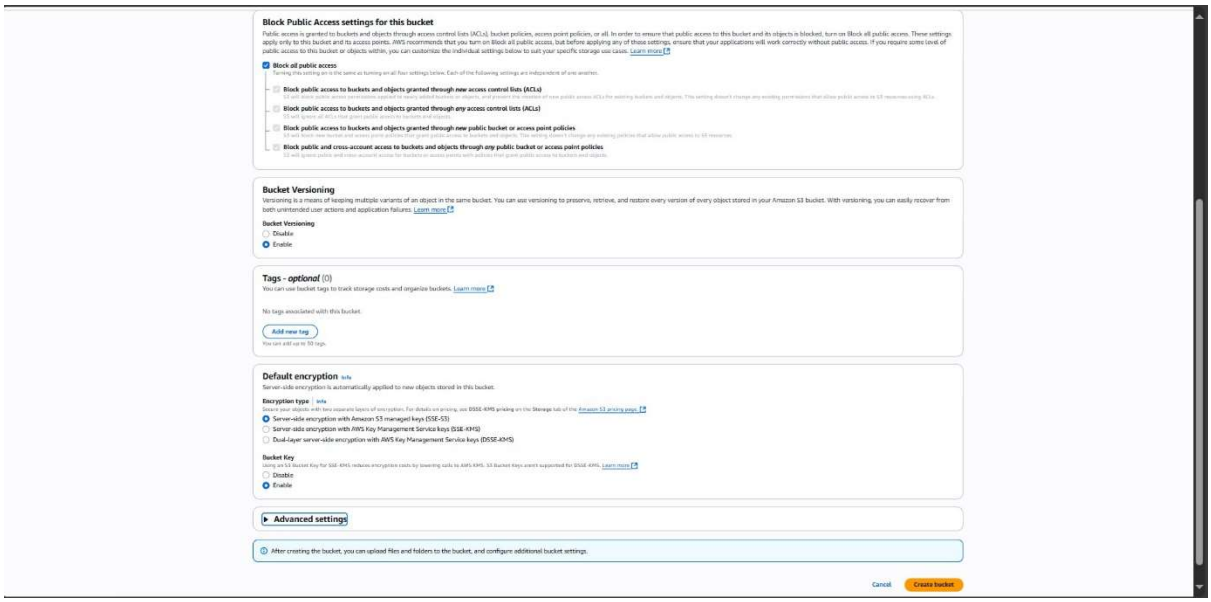
By using CloudFront, you can dramatically improve your website's performance, making users happier and potentially boosting your application and business.

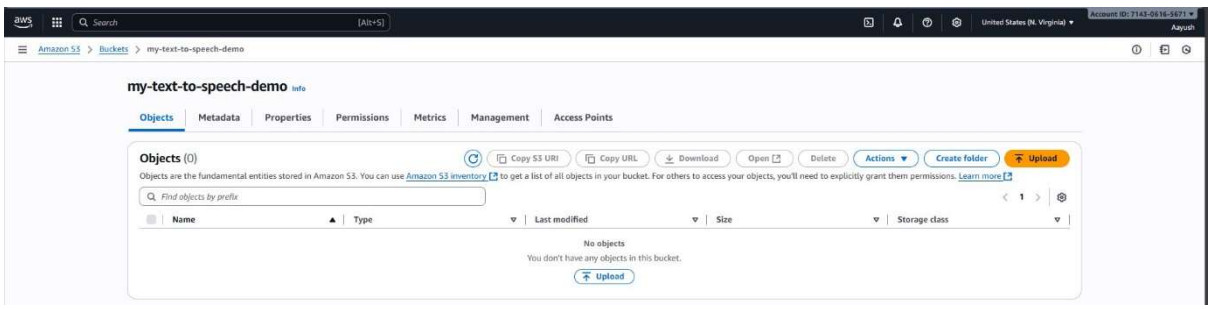
Project

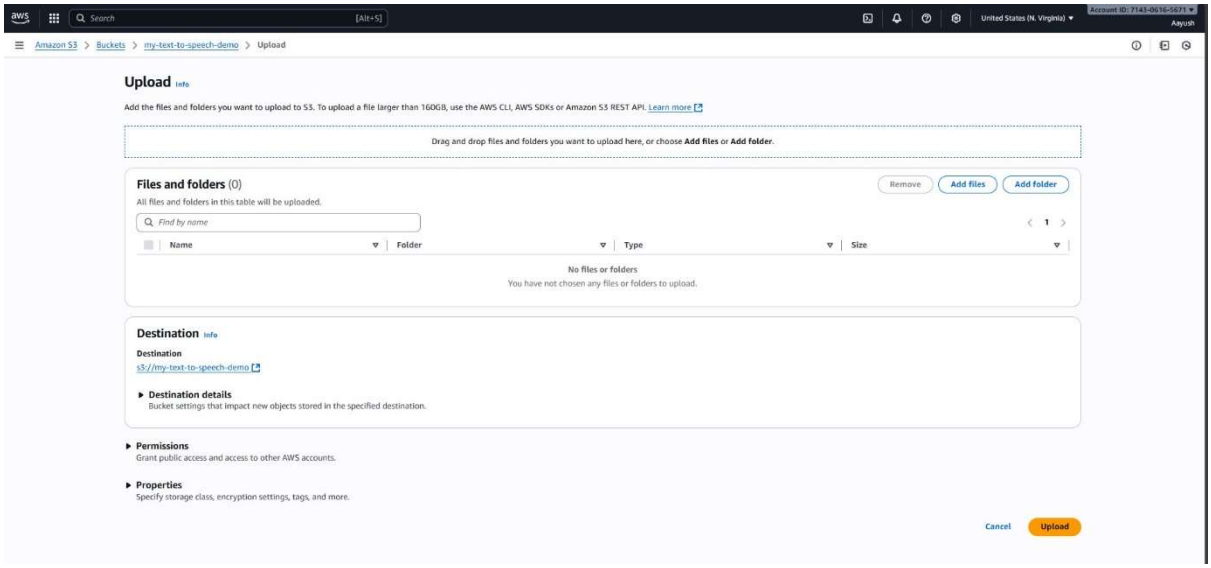
Deploying Static web site using S3 and CloudFront(CDN)

Step 1: - Create a S3 bucket and upload you website into it.

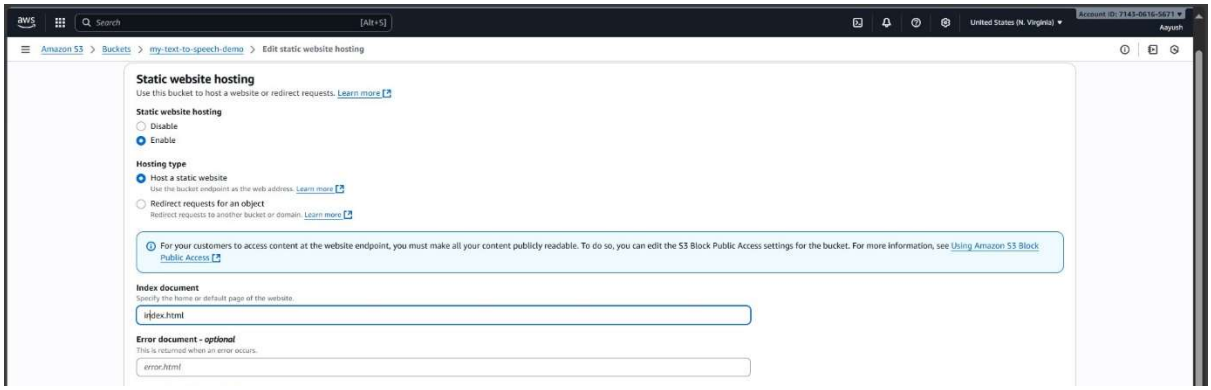






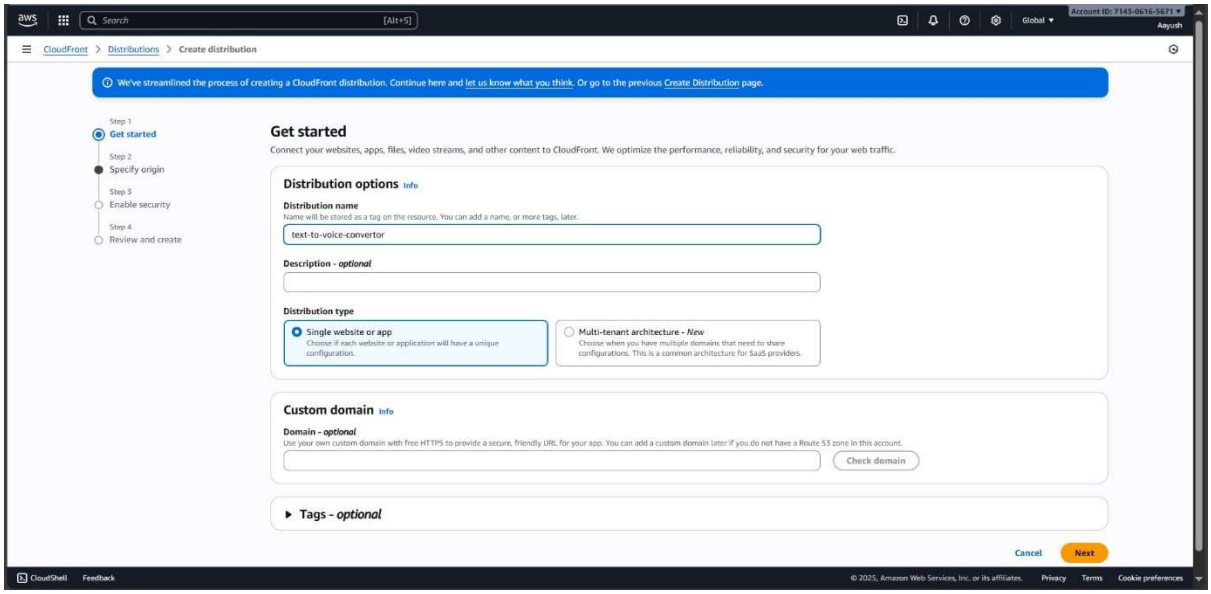


Step 3: - Now goto properties section and enable the static website hosting, select hosting type and enter index document name.

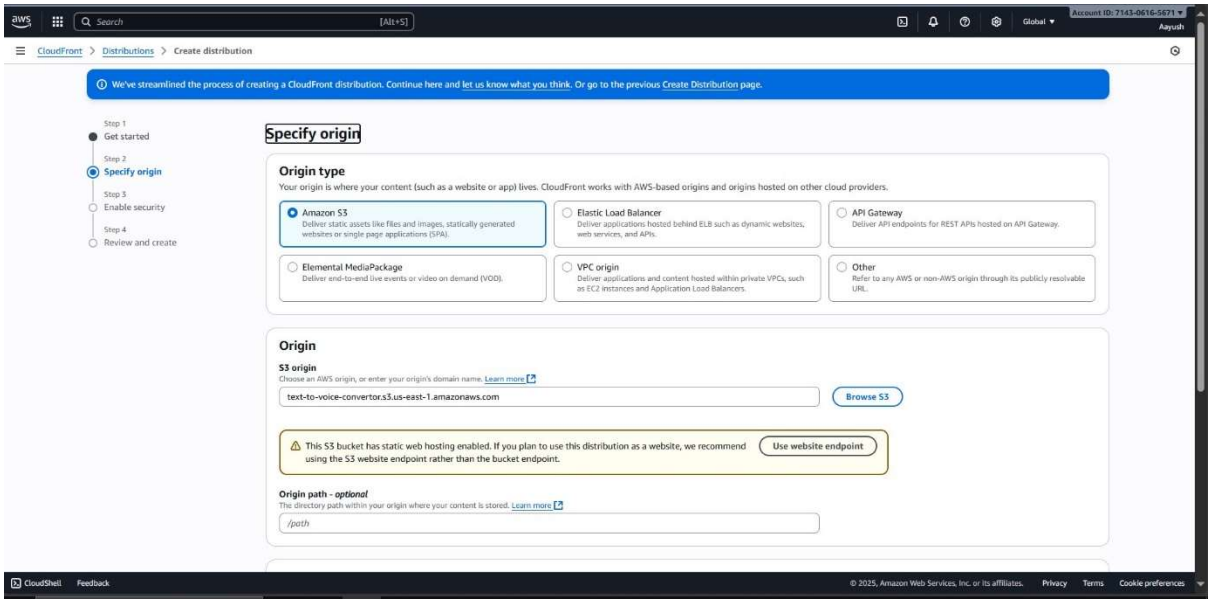


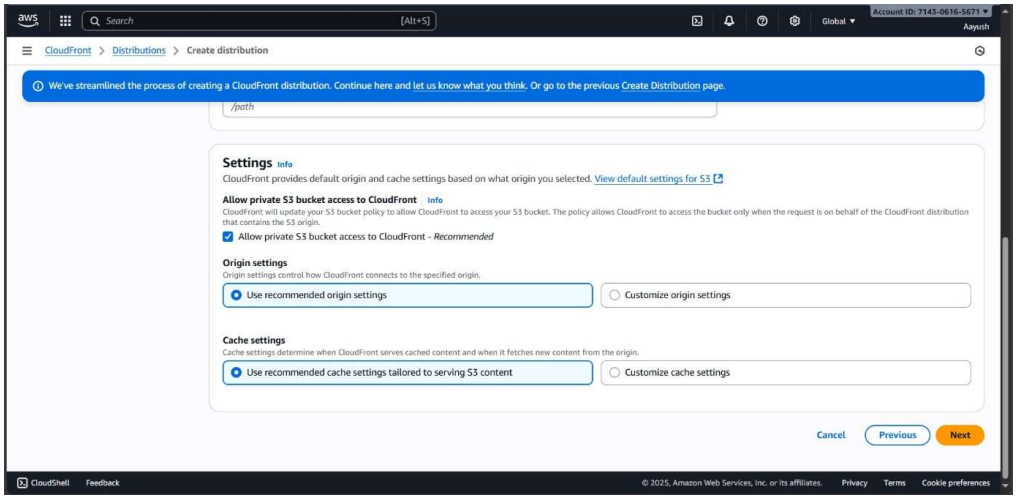
Step 4: - Now Search for CloudFront and goto its dashboard and click on Create Distribution, enter distribution name and

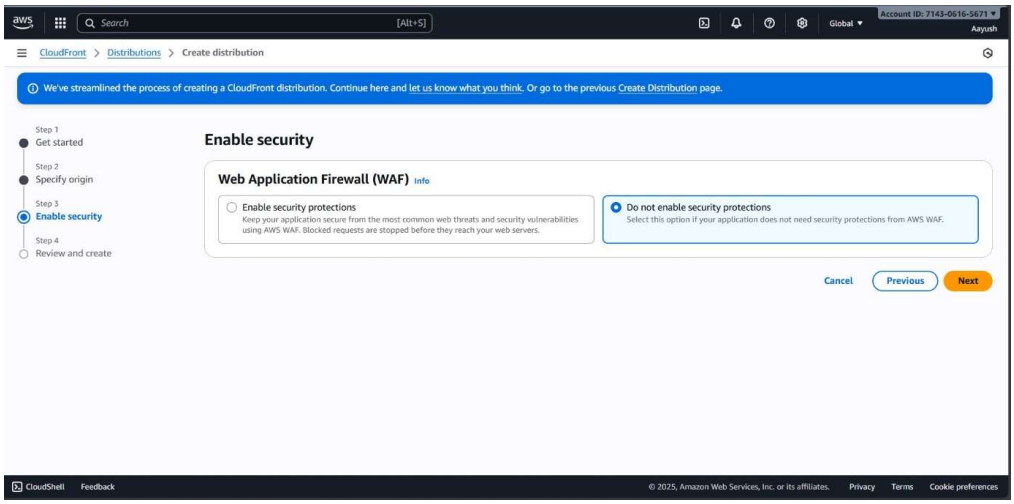
select its type and click on next.

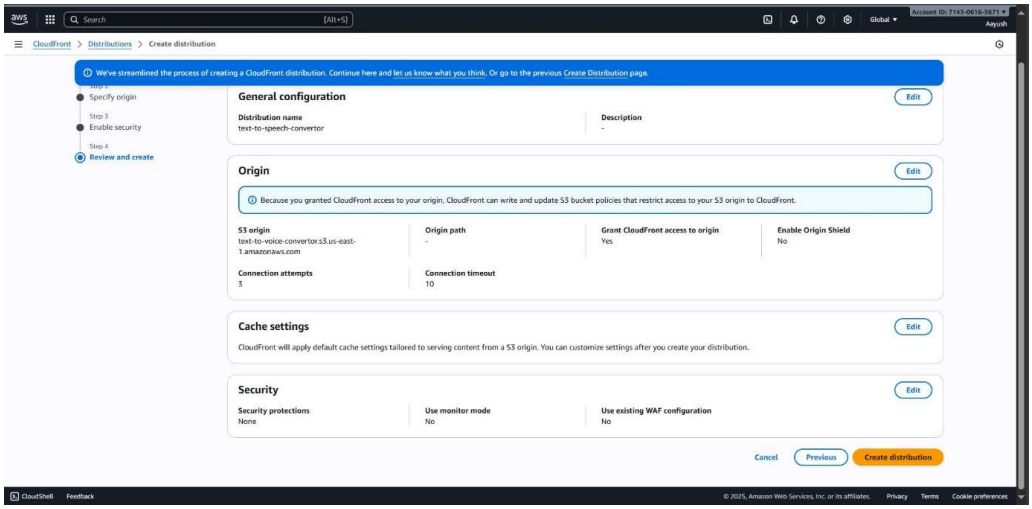


Step 5: - Now select origin type, origin file and click on next and then select WAF accordingly. (do not enable for demo purpose)

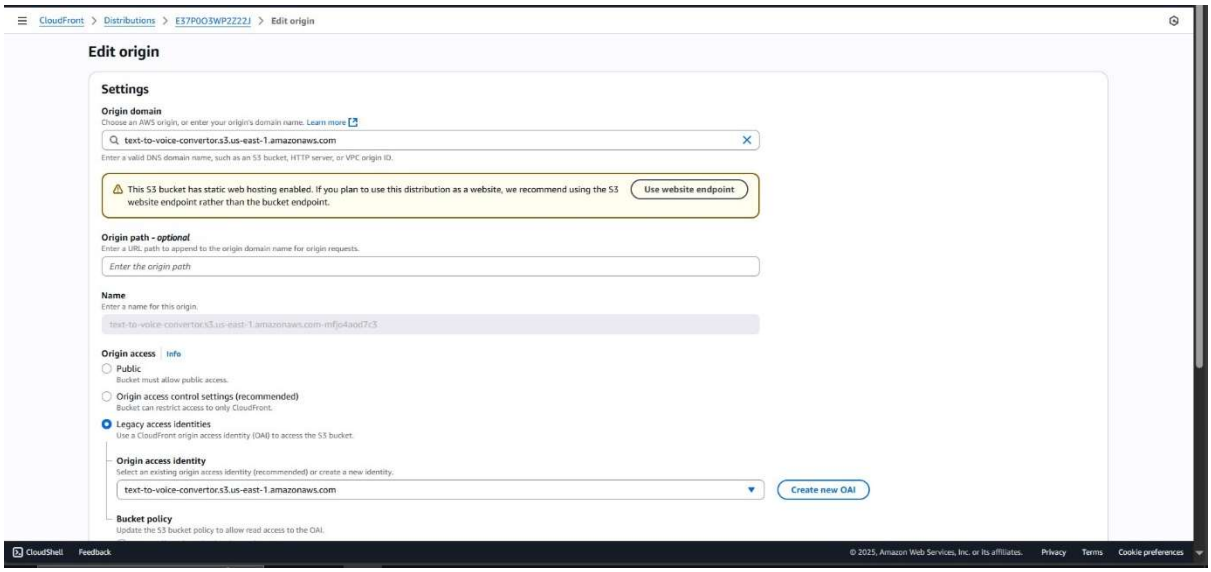


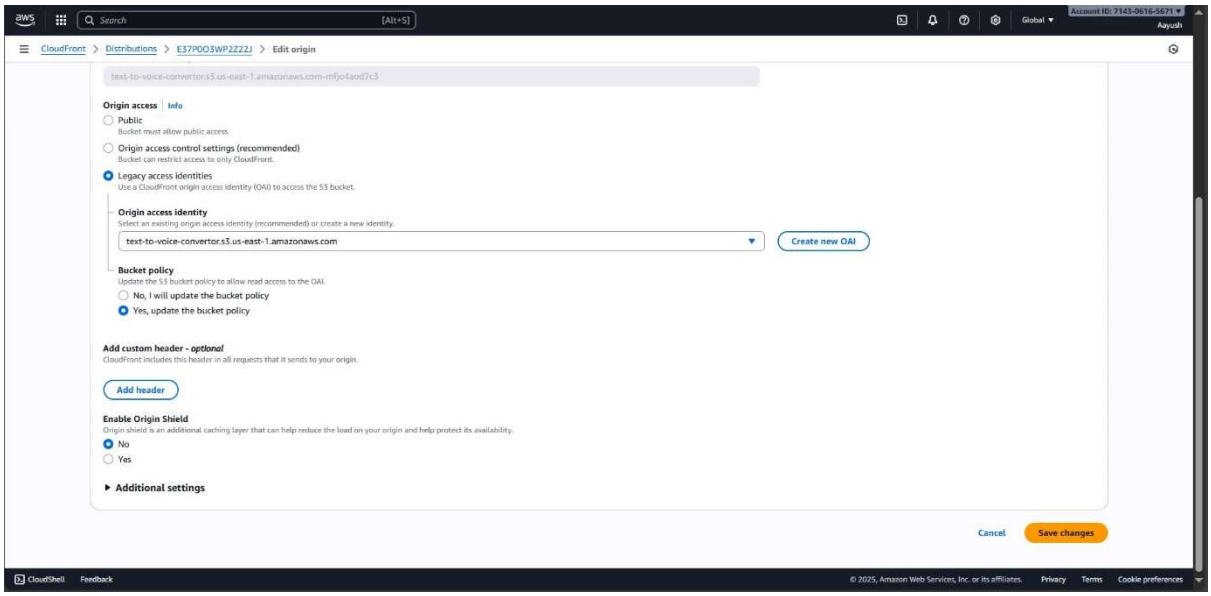

Search [Alt+S]
Global
Account ID: 7143-0616-5671
Aayush
CloudFront > Distributions > Create distribution
We've streamlined the process of creating a CloudFront distribution. Continue here and let us know what you think. Or go to the previous Create Distribution page.
/path
Settings Info
CloudFront provides default origin and cache settings based on what origin you selected. View default settings for S3
Allow private S3 bucket access to CloudFront Info
CloudFront will update your S3 bucket policy to allow CloudFront to access your S3 bucket. The policy allows CloudFront to access the bucket only when the request is on behalf of the CloudFront distribution that contains the S3 origin.
Allow private S3 bucket access to CloudFront - Recommended
Origin settings
Origin settings control how CloudFront connects to the specified origin.
Use recommended origin settings
Customize origin settings
Cache settings
Cache settings determine when CloudFront serves cached content and when it fetches new content from the origin.
Use recommended cache settings tailored to serving S3 content
Customize cache settings
Cancel
Previous
Next
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Search [Alt+S]
Global
Account ID: 7143-0616-5671
Aayush
CloudFront > Distributions > Create distribution
We've streamlined the process of creating a CloudFront distribution. Continue here and let us know what you think. Or go to the previous Create Distribution page.
Step 1
Get started
Step 2
Specify origin
Step 3
Enable security
Step 4
Review and create
Enable security
Web Application Firewall (WAF) Info
Enable security protections
Keep your application secure from the most common web threats and security vulnerabilities using AWS WAF. Blocked requests are stopped before they reach your web servers.
Do not enable security protections
Select this option if your application does not need security protections from AWS WAF.
Cancel
Previous
Next
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

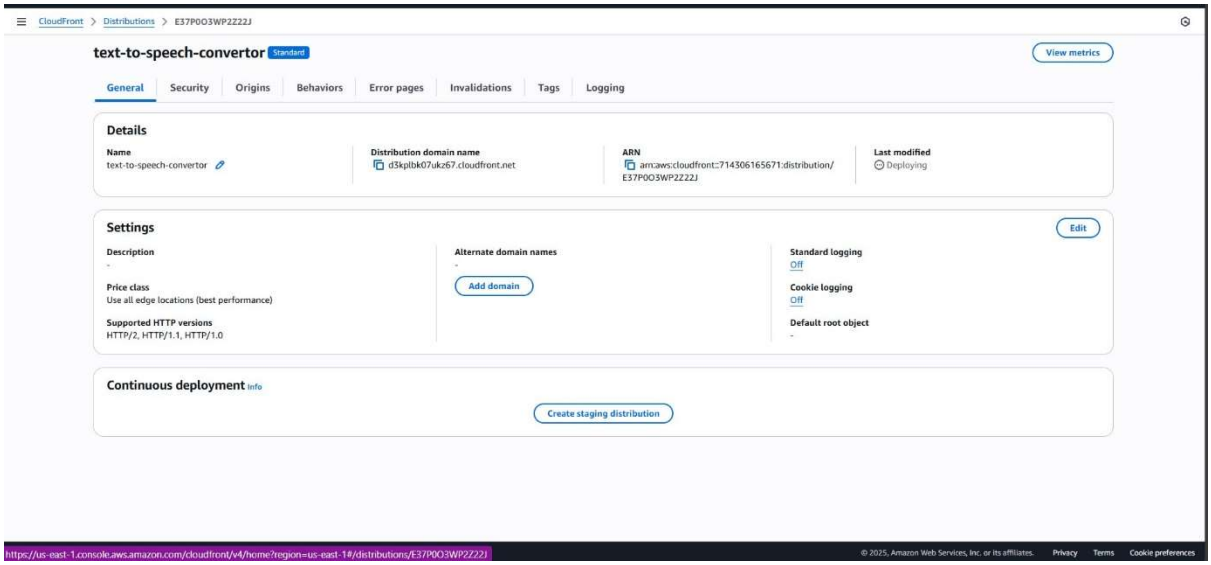

Search [Alt+S]
Global
Account ID: 7143-0616-5671
Aayush
CloudFront > Distributions > Create distribution
We've streamlined the process of creating a CloudFront distribution. Continue here and let us know what you think. Or go to the previous Create Distribution page.
Specify origin
Enable security
Review and create
General configuration
Edit
Distribution name
text-to-speech-convertor
Description
-
Origin
Edit
Because you granted CloudFront access to your origin, CloudFront can write and update S3 bucket policies that restrict access to your S3 origin to CloudFront.
S3 origin
text-to-voice-convertor.s3.us-east-1.amazonaws.com
Origin path
-
Grant CloudFront access to origin
Yes
Enable Origin Shield
No
Connection attempts
3
Connection timeout
10
Cache settings
Edit
CloudFront will apply default cache settings tailored to serving content from a S3 origin. You can customize settings after you create your distribution.
Security
Edit
Security protections
None
Use monitor mode
No
Use existing WAF configuration
No
Cancel
Previous
Create distribution
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

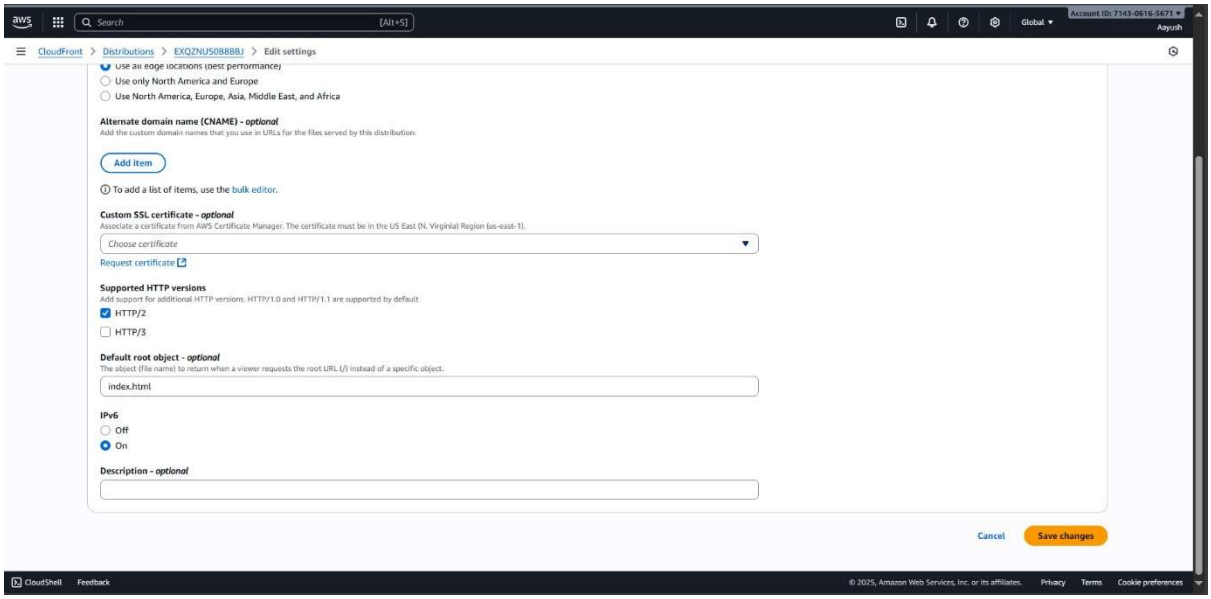
Step 6: - Now click on distribution name and click on origin section and edit it and create a OAI and click on save changes.



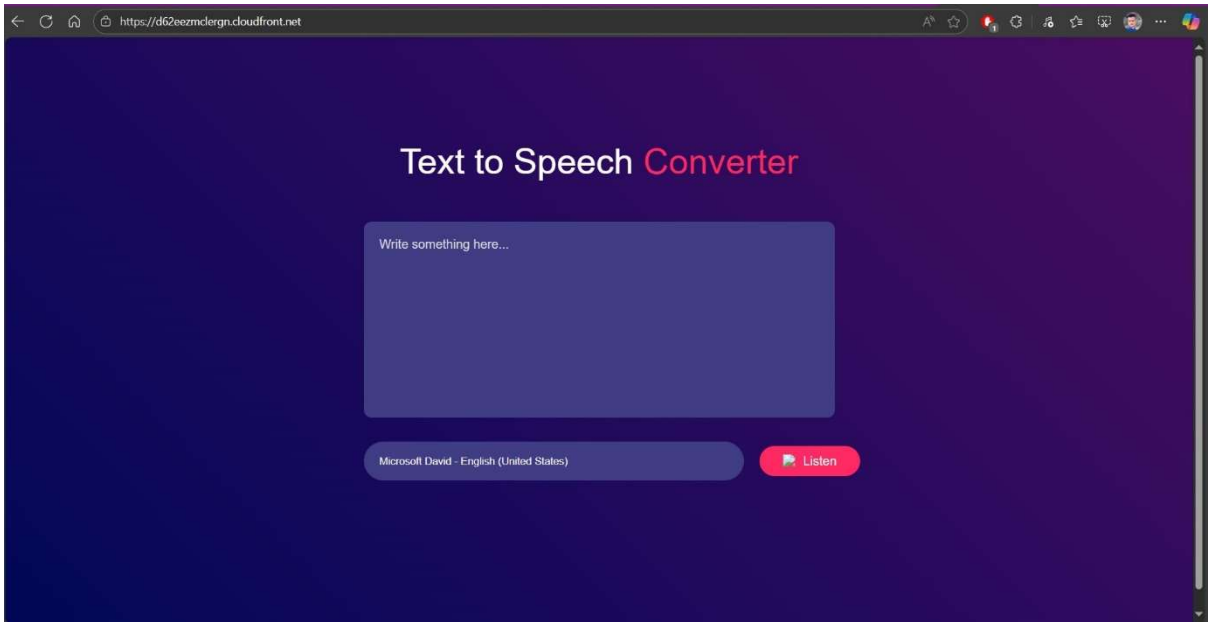


Step 7: - Now on general section click on edit and add default root object and save the change and wait for depolyement ready it will take some time.





Step 8: - Now copy the distribution domain name and paste it on your browser.



Note: - Must delete all the resources after demonstration.

AWS Elastic Container Registry (ECR)

AWS Elastic Container Registry (ECR) is a fully managed container image registry service provided by Amazon Web Services (AWS). It enables you to store, manage, and deploy container images (Docker images) securely, making it an essential component of your containerized application development workflow. ECR integrates seamlessly with other AWS services like **Amazon Elastic Container Service (ECS)** and **Amazon Elastic Kubernetes Service (EKS).**

Key Benefits of ECR

- **Security:** ECR offers encryption at rest, and images are stored in private repositories by default, ensuring the security of your container images.
- **Integration:** ECR integrates smoothly with AWS services like ECS and EKS, simplifying the deployment process.
- **Scalability:** As a managed service, ECR automatically scales to meet the demands of your container image storage.
- **Availability:** ECR guarantees high availability, reducing the risk of image unavailability during critical times.
- **Lifecycle Policies:** You can define lifecycle policies to automate the cleanup of unused or old container images, helping you save on storage costs.

Components of Amazon ECR

Registry

An Amazon ECR registry is a private repository provided to each AWS account, where you can create one or more repositories. These repositories allow you to store and distribute Docker images, Open Container Initiative (OCI) images, and other OCI-compatible artifacts within your AWS environment.

Authorization token

Your client must authenticate to an Amazon ECR private registry as an AWS user before it can push and pull images.

Repository

A repository in Amazon ECR is a logical collection where you can store your Docker images, Open Container Initiative (OCI) images, and other OCI-compatible artifacts. Within a single Amazon ECR registry, you can have multiple repositories to organize your container images.

Repository policy

You can control access to your repositories and the contents within them with repository policies.

Image

You can push and pull container images to your repositories. You can use these images locally on your development system, or you can use them in Amazon ECS task definitions and Amazon EKS pod specifications.

Lifecycle Policy

Amazon ECR lifecycle policies allow you to manage the lifecycle of your images by defining rules for pruning and expiring old or unused images.

Image Scanning

Amazon ECR provides an integrated image scanning capability that helps identify software vulnerabilities in your container images.

Access Control

Amazon ECR uses IAM to control access to your repositories. You can create IAM users, groups, and roles with specific permissions to push, pull, or manage Amazon ECR repositories.

Cross-account and Cross-region Replication

Amazon ECR supports replicating images across multiple AWS accounts and regions for increased availability and reduced latency.

Encryption

Amazon ECR supports server-side encryption of your Docker images at rest using AWS KMS.

AWS Command Line Interface Integration

The AWS CLI provides commands to interact with Amazon ECR repositories, such as creating, listing, pushing, and pulling images.

AWS Management Console

Amazon ECR can also be managed through the AWS Management Console, providing a user-friendly web interface for working with your repositories and images.

Amazon CloudWatch

Amazon ECR provides metrics and logs that can be monitored using Amazon CloudWatch, enabling you to track the performance and usage of your Amazon ECR repositories.

Getting Started with AWS ECR

1. Creating an ECR Repository

1. Go to the AWS Management Console and navigate to the Amazon ECR service.
2. Click on "Create repository" to create a new repository.
3. Enter a unique name for your repository and click "Create repository."

2. Installing AWS CLI

To interact with ECR from your local machine, you'll need to have the AWS Command Line Interface (CLI) installed. Follow the instructions in the AWS CLI User Guide to install it.

3. Configuring AWS CLI

After installing the AWS CLI, open a terminal and run the following command to configure your CLI with your AWS credentials:

aws configure

Enter your AWS Access Key ID, Secret Access Key, default region, and preferred output format when prompted.

Pushing Docker Images to ECR

Now that you have your ECR repository set up and the AWS CLI configured, let's push a Docker image to ECR.

1. Build your Docker image locally using the docker build command:

docker build -t <your-image-name> <path-to-dockerfile>

2. Tag the image with your ECR repository URI:

docker tag <your-image-name>:<tag> <your-aws-account-id>.dkr.ecr.<your-region>.amazonaws.com/<your-repository-name>:<tag>

3. Log in to your ECR registry using the AWS CLI:

aws ecr get-login-password --region <your-region> | docker login --username AWS --password-stdin <your-aws-account-id>.dkr.ecr.<your-region>.amazonaws.com

4. Push the Docker image to ECR:

docker push <your-aws-account-id>.dkr.ecr.<your-region>.amazonaws.com/<your-repository-name>:<tag>

Pulling Docker Images from ECR

To pull and use the Docker images from ECR on another system or AWS service, follow these steps:

1. Log in to ECR using the AWS CLI as shown in Step 3 of the previous section.
2. Pull the Docker image from ECR:

docker pull <your-aws-account-id>.dkr.ecr.<your-region>.amazonaws.com/<your-repository-name>:<tag>

Cleaning Up Resources

As good practice, remember to clean up resources that you no longer need to avoid unnecessary costs.