CLOUD FRONT DISTRIBUTION

Prepared by Lalit Kumar

linkedin.com/in/lalit192977

What is AWS CloudFront?

CloudFront is a **Content Delivery Network (CDN)** service provided by AWS. Its main job is to deliver **content (websites, images, videos, APIs, files)** to users **faster, securely, and with low latency**.

👉 Imagine your server is in the US, but a user is opening your site from India.
Without CloudFront → the request goes all the way to the US → slow response.
With CloudFront → AWS stores (caches) your content in **Edge Locations (global data centers).** The Indian user gets content from the nearest edge (like Mumbai or Singapore) → much faster.

How CloudFront Works (Simple Flow)

1. User requests your website → goes to CloudFront.
2. CloudFront checks:
    - If content is cached at the nearest edge → it sends immediately.
    - If not → it fetches from your Origin (S3, EC2, Load Balancer, API Gateway, etc.), caches it, then sends to the user.
3. Next user request from that region → gets super-fast response from the cache.

Common Use Cases

1. Static Website Hosting – Store files in S3 + serve via CloudFront for global reach.
2. Video Streaming – Used by platforms like Netflix/Prime for smooth streaming.
3. Dynamic Content Acceleration – Speeds up APIs, dashboards, logins.

4. E-Commerce – Faster loading of product images, catalogs, and checkout pages.
5. Software Downloads – Ideal for large file delivery (games, apps, OS updates).
6. Security – Integrates with AWS WAF & Shield for protection against DDoS and attacks.

Why Use CloudFront? (Benefits)

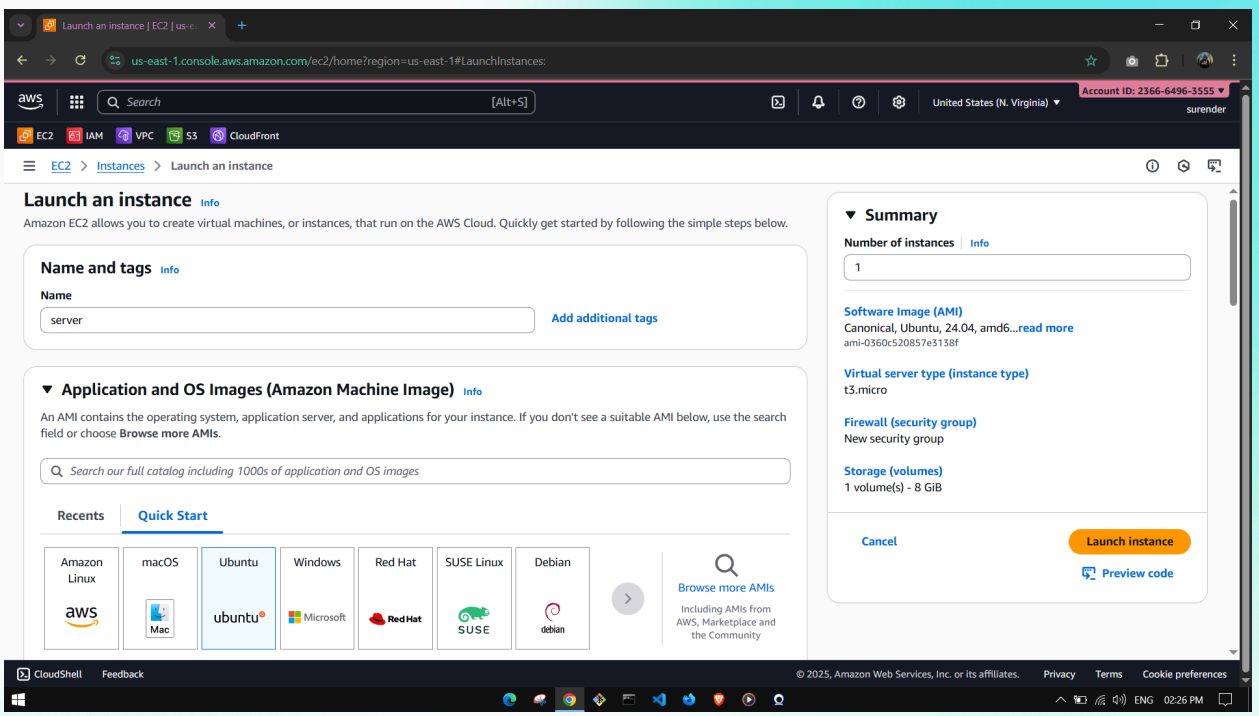
- 🌍 Global Edge Network → users get content from the nearest AWS location.
- ⚡ High Performance → cached data makes repeat requests super fast.
- 💰 Cost Effective → reduces load on your servers, saving bandwidth + compute cost.
- 🔒 Secure → HTTPS, custom SSL, WAF, geo-restrictions, DDoS protection.
- 📈 Scalable → handles sudden traffic spikes automatically.

Where to Use CloudFront?

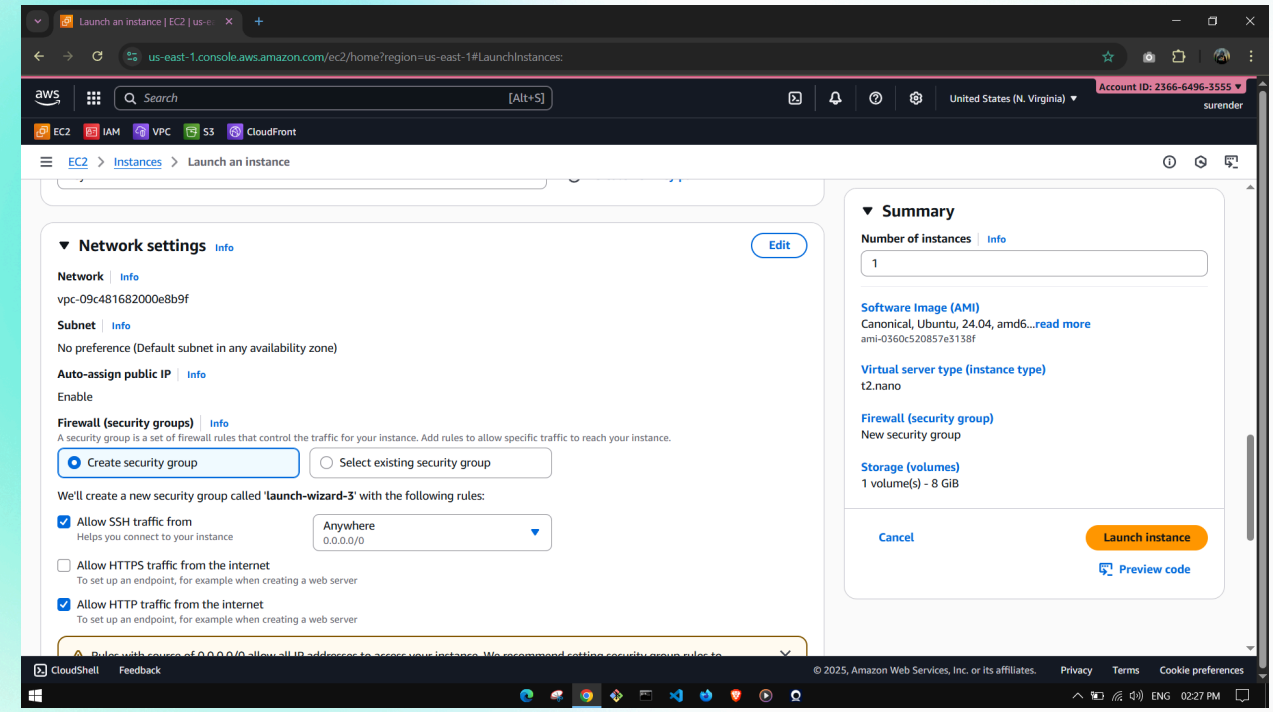
- When your website/app serves a global audience.
- For high-traffic sites where you don't want to overload your servers.
- On image-heavy / video-heavy platforms.
- For E-commerce websites needing fast + secure checkout experience.
- To accelerate APIs used in mobile or SaaS apps.

Let's do practice

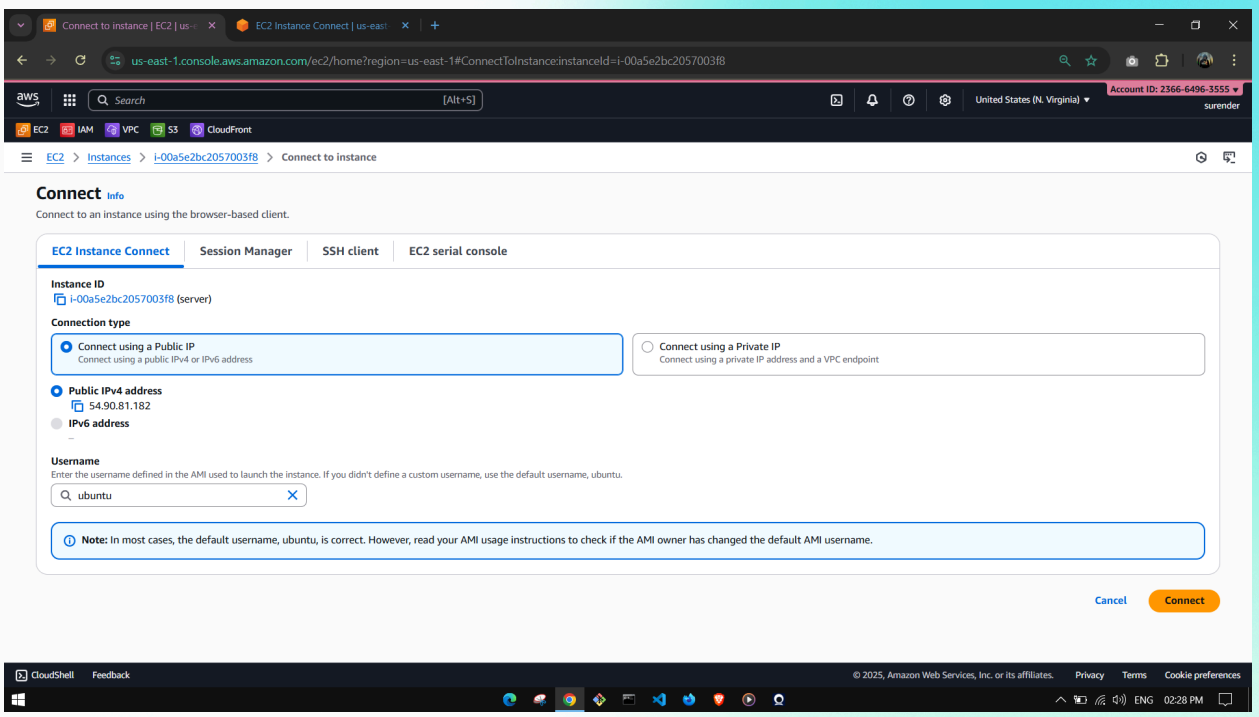
Lets starts withe the ec2 instance server, give name to the instance and choose AMI here we choose ubuntu



From the network settings create a security group for this server allowing http (port-80) and ssh (port-22)



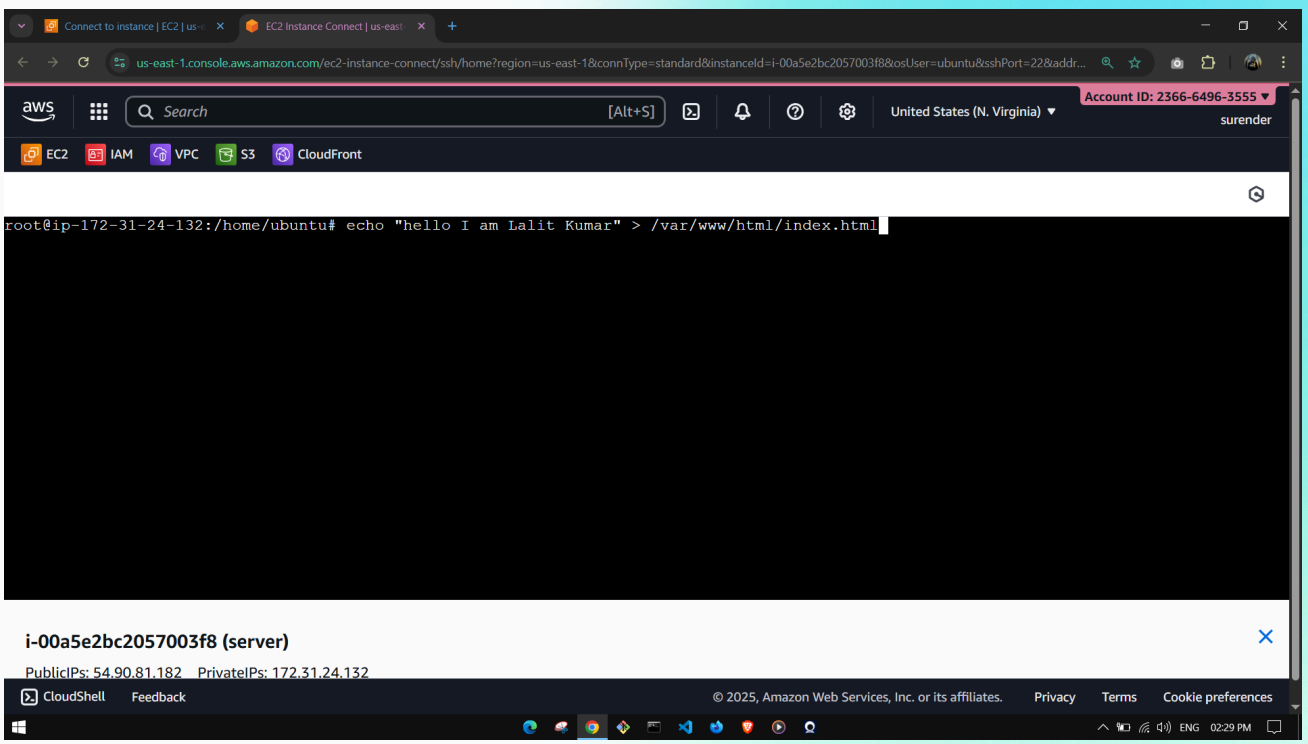
Connect your server



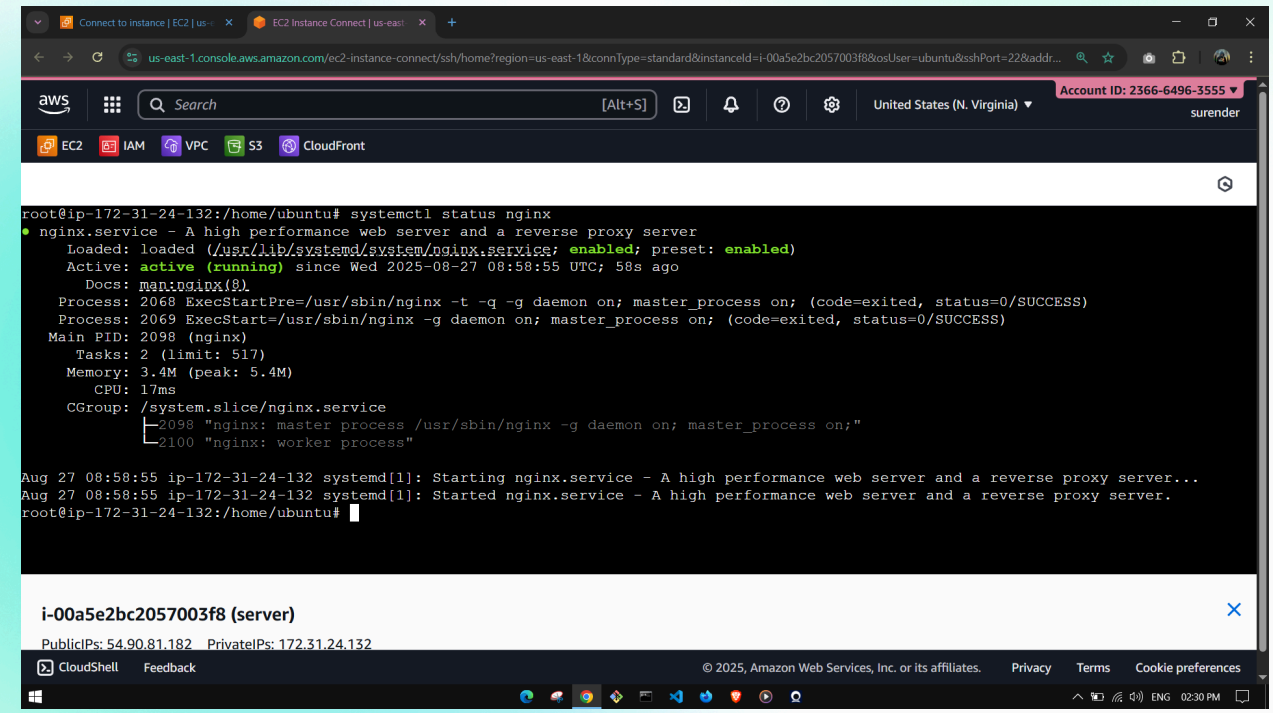
After connecting it update it and install nginx server through the below commands-

apt update and after that run apt install nginx -y

Now configure your nginx server

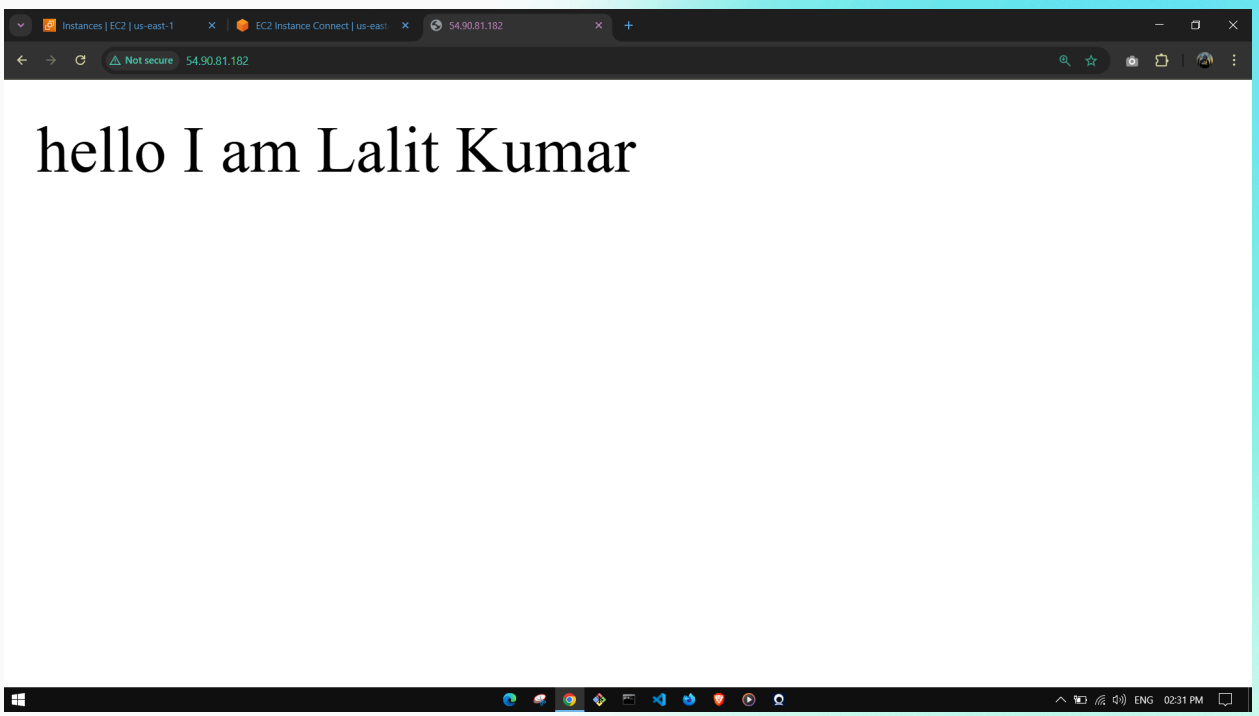


Check the nginx server is running or not shown in the below image if the nginx server is not running then run it through **systemctl start nginx**

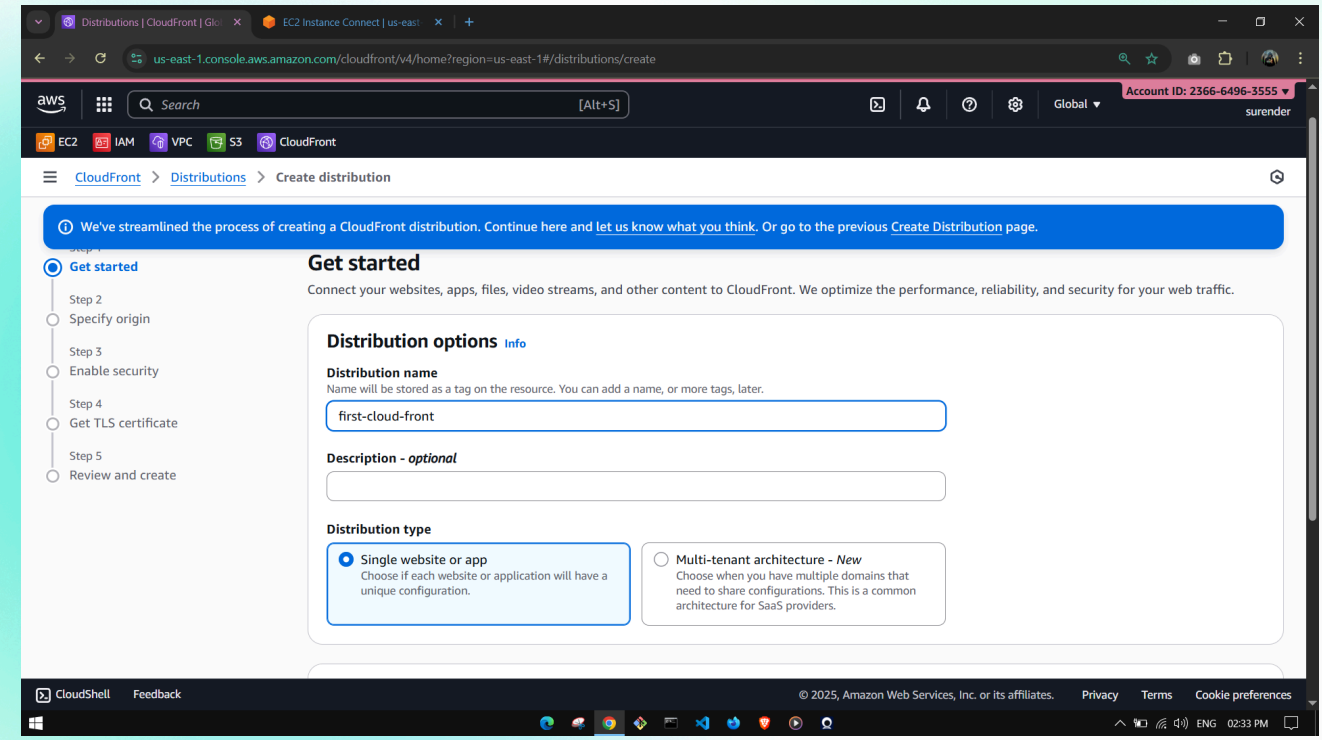


Now select your instance and copy the public ip address/public dns and paste it in your browser and see your web page is running or not.

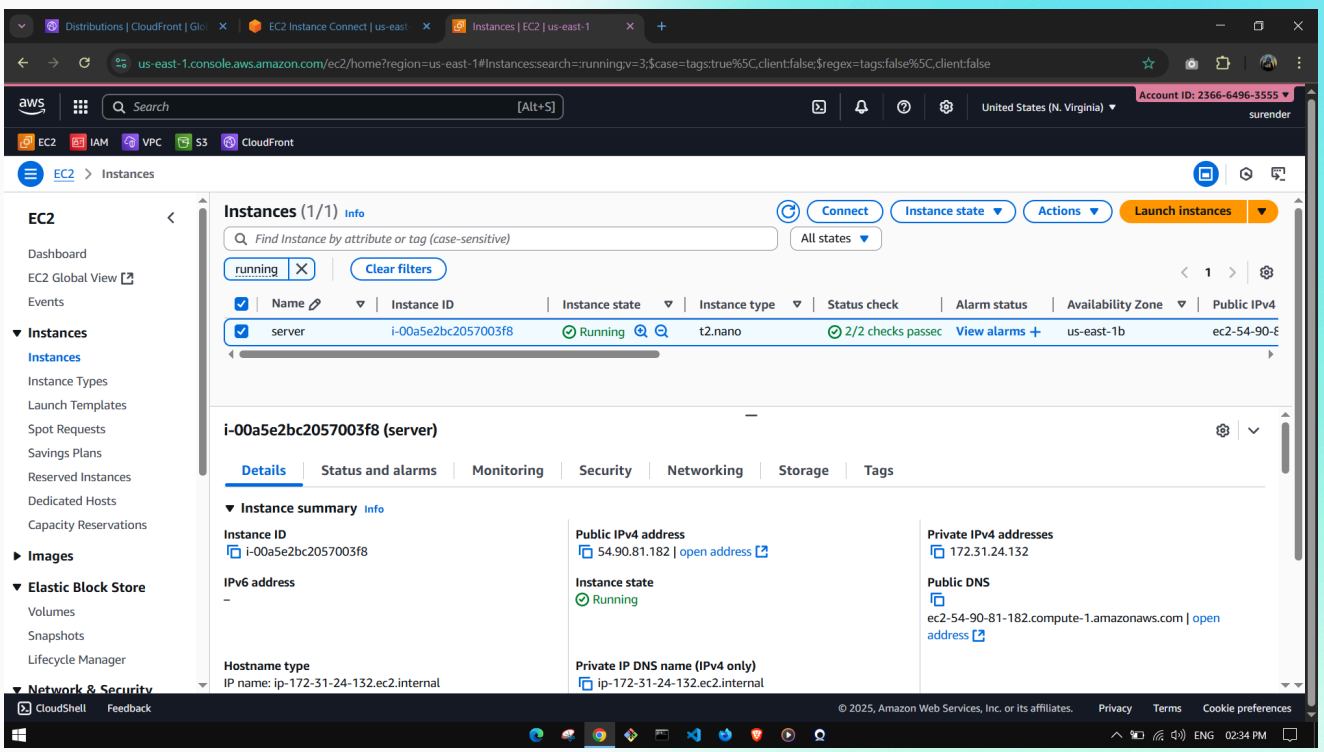
Just check here...



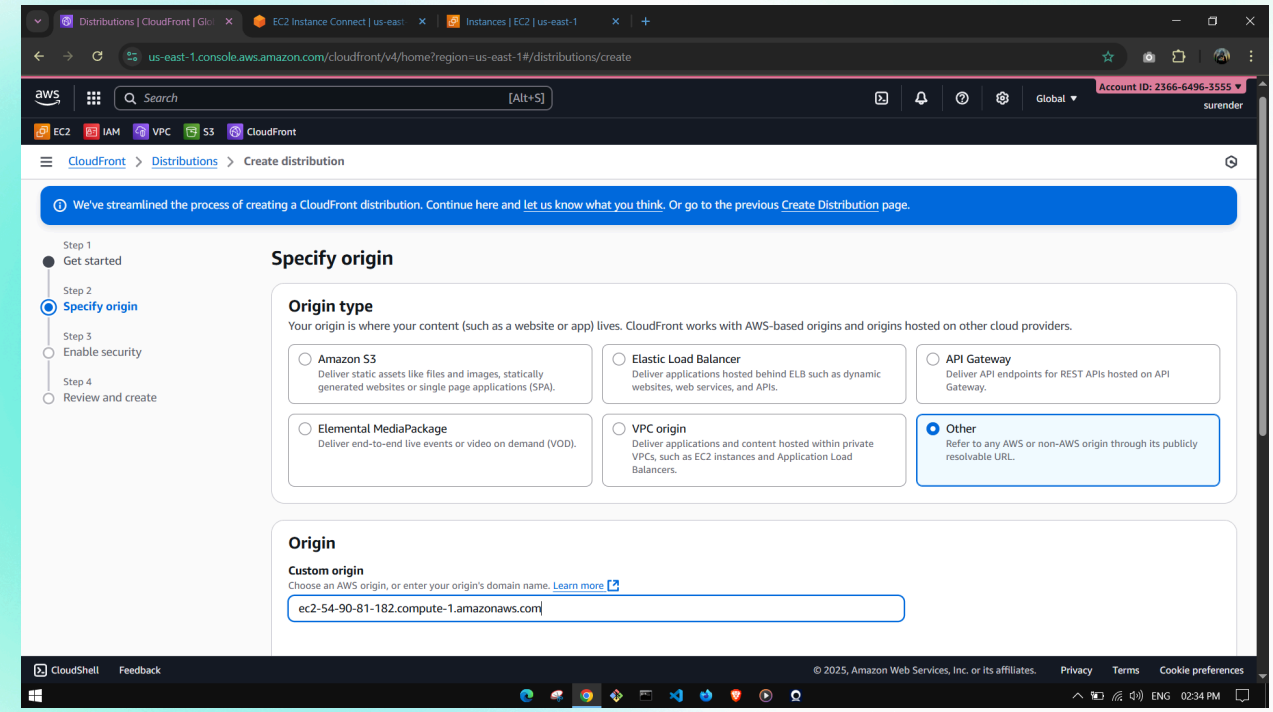
Now go to cloudfront service and create a cloudfront template, give the name and for now select single website or app from the distribution type to make it simple.



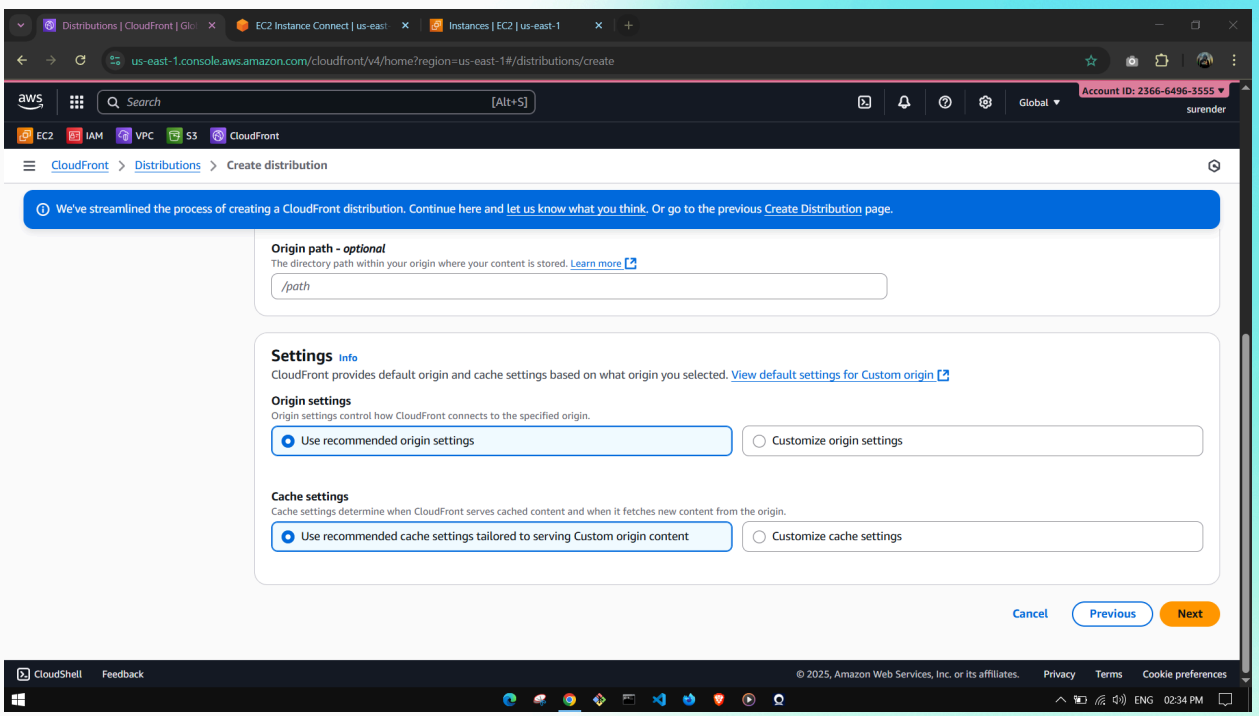
Go to your instance and copy your instance DNS



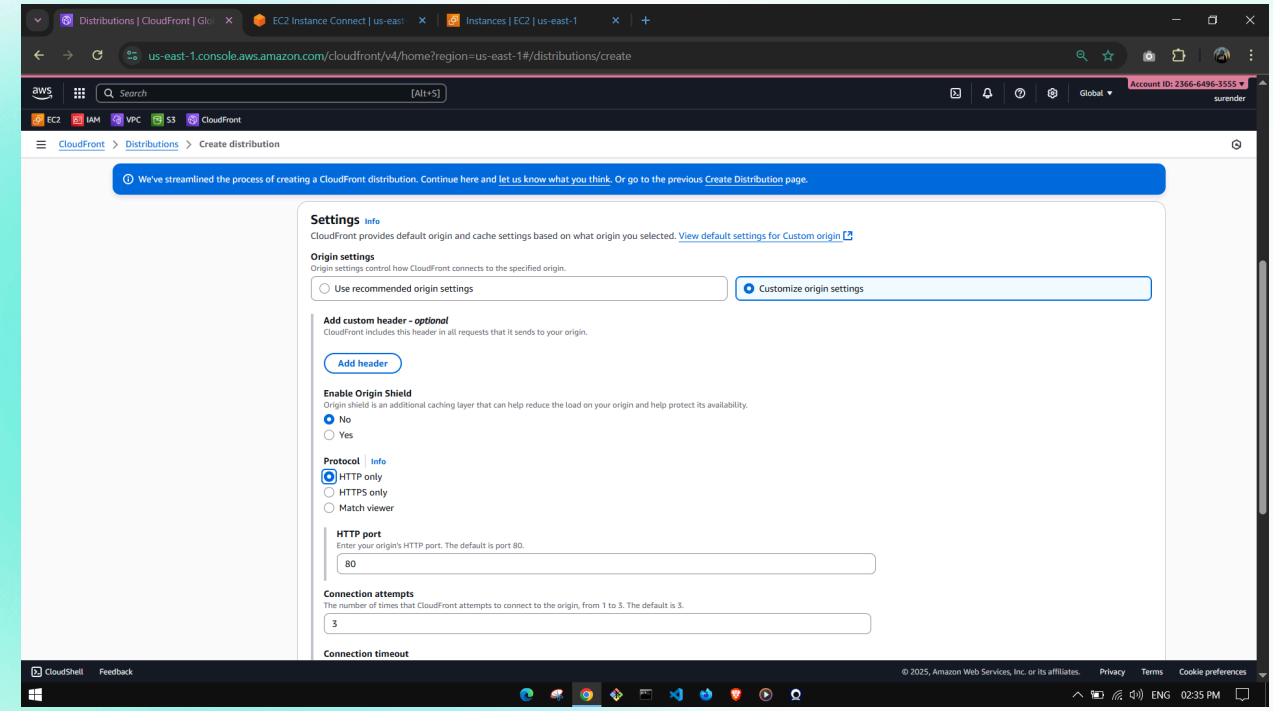
You have different types of origin types for your cloud distribution, but now select another and paste your ec2 instance DNS as origin and go for the next step.



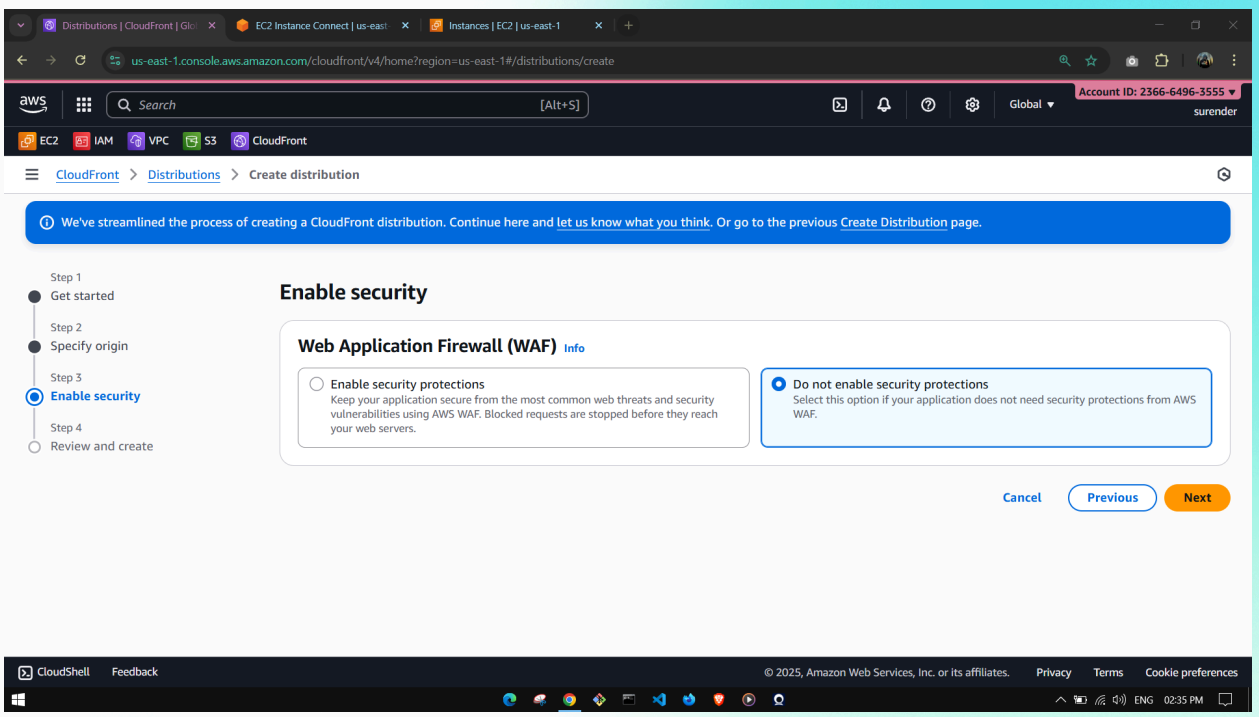
From the settings choose custom origin settings



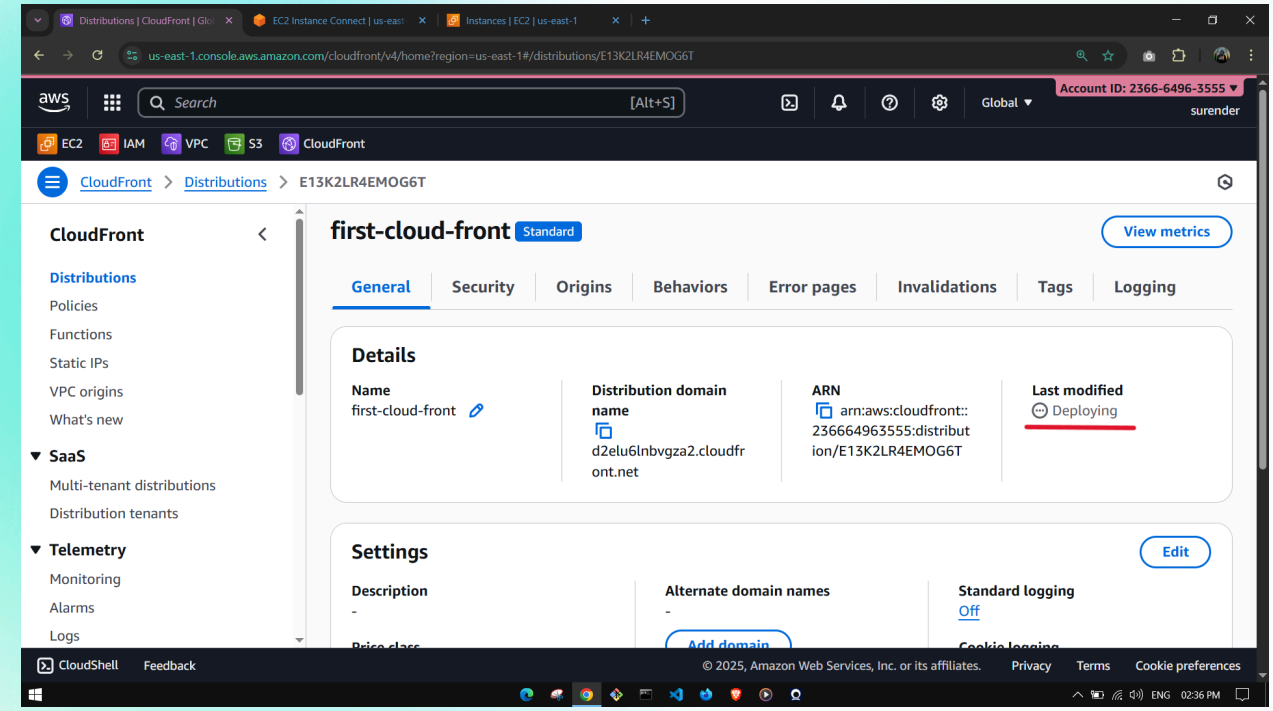
Select HTTP only and go for next step



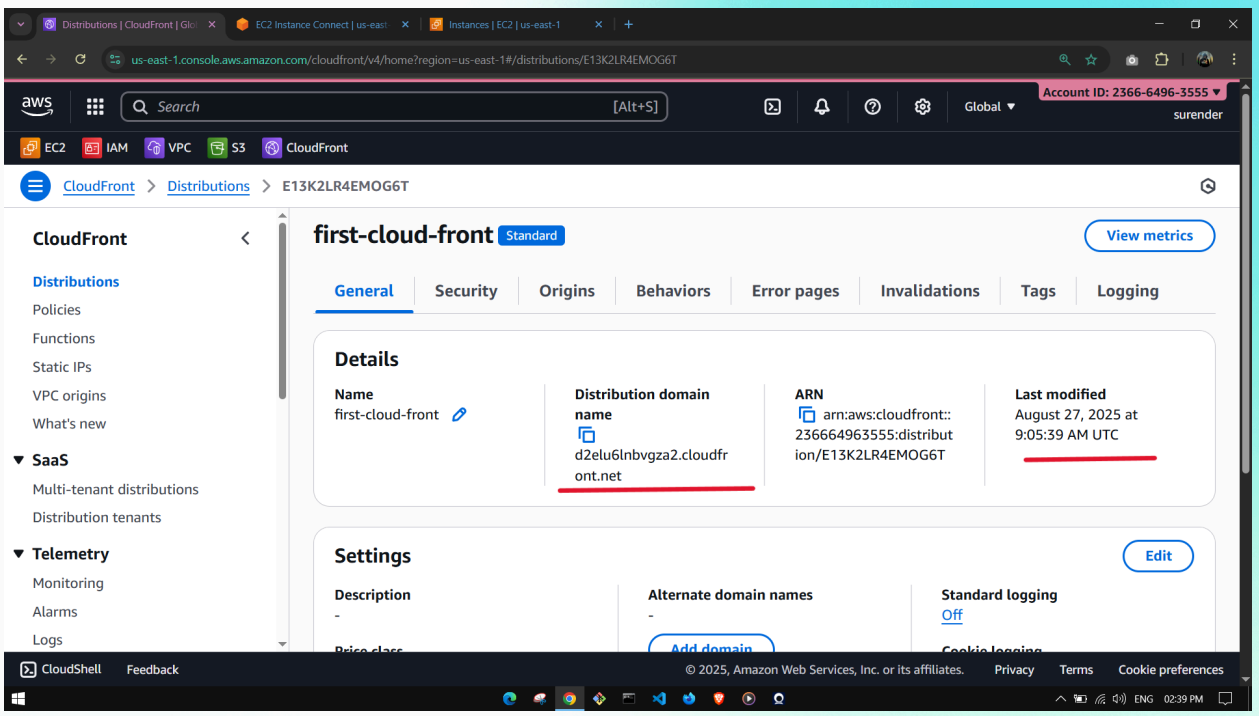
For now just disable the web application firewall (WAF) and go for the next step and create a cloud formation template.



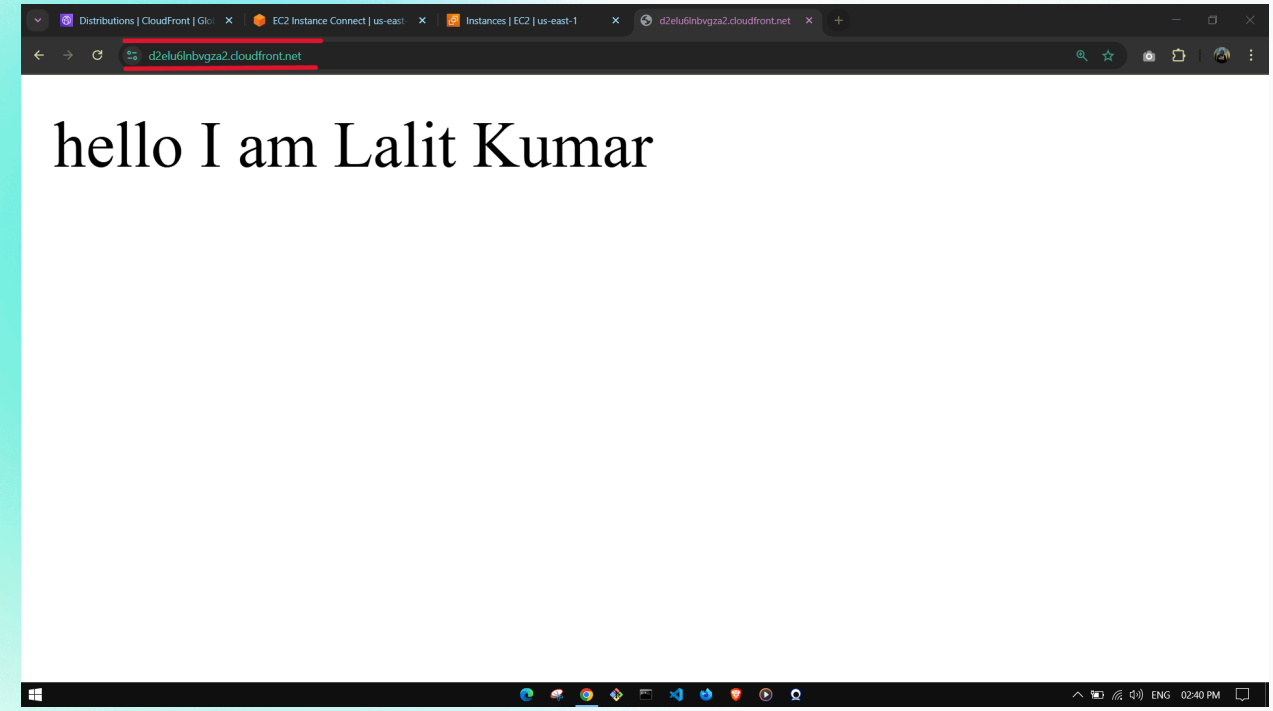
Wait for the deployment it will take some time to deploy your application id different aws edge location so it may take time.



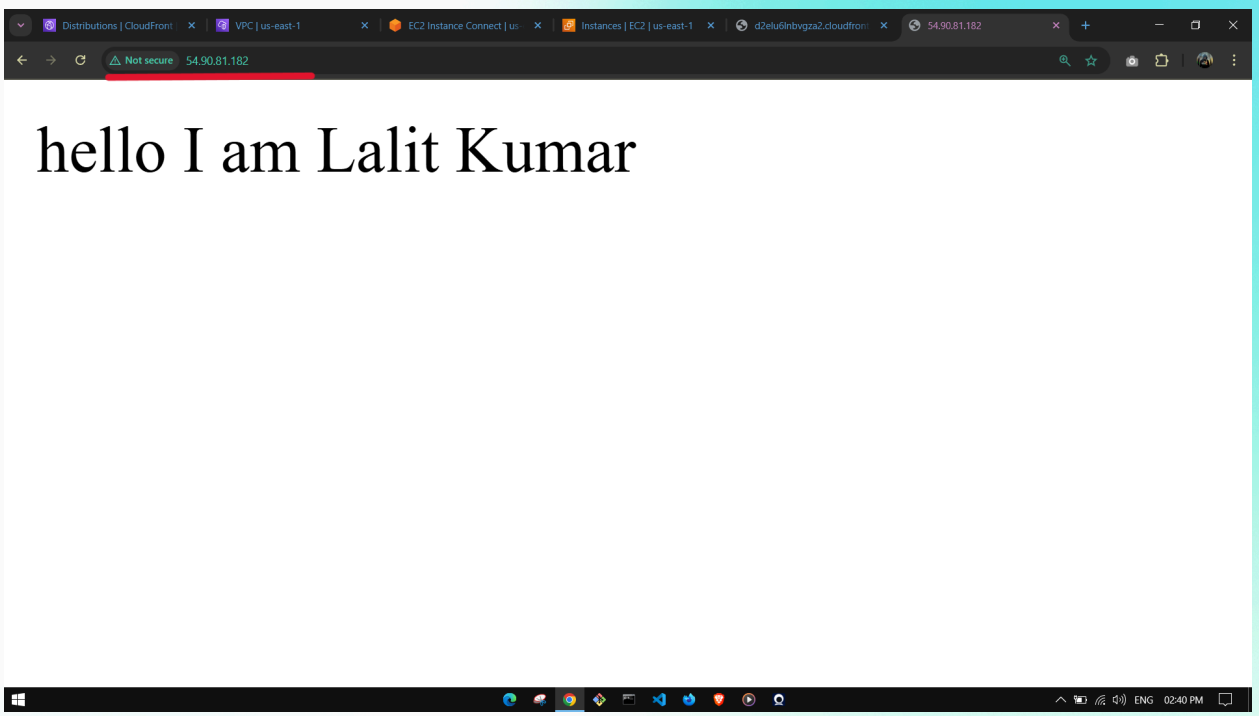
After the deployment copy the DNS and paste it in your browser



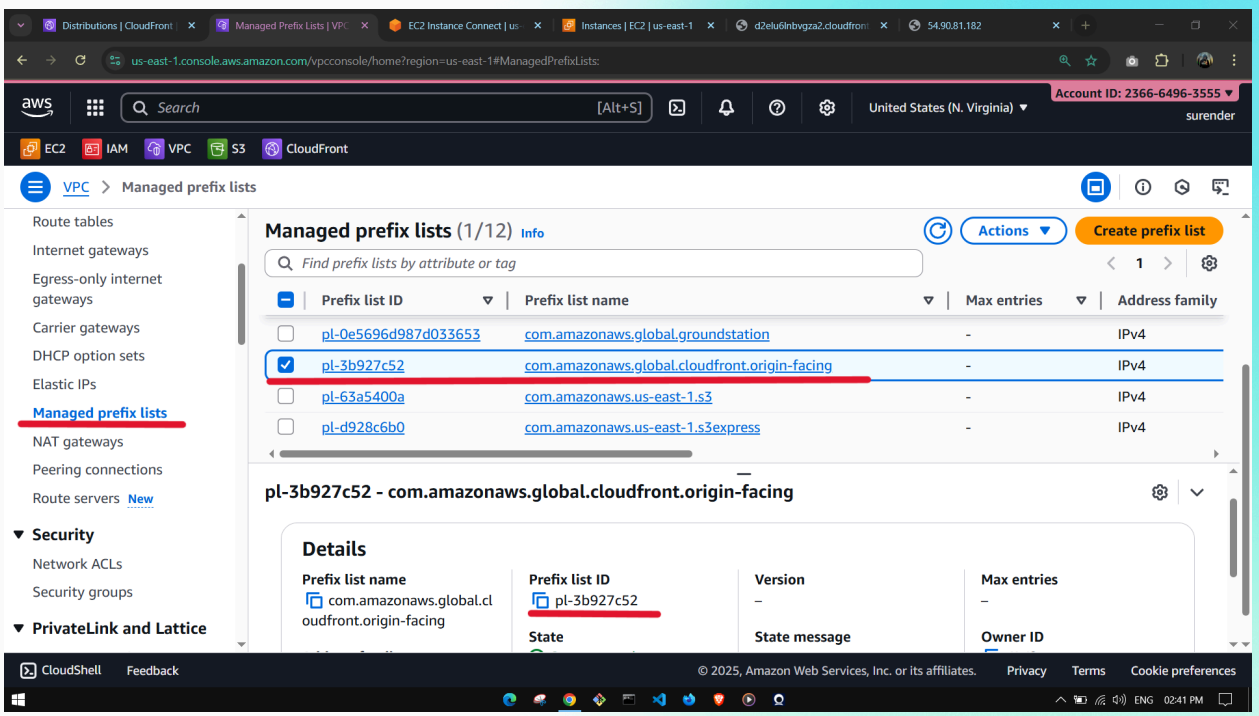
Here you can see your web-application is accessible through cloudfront you created.



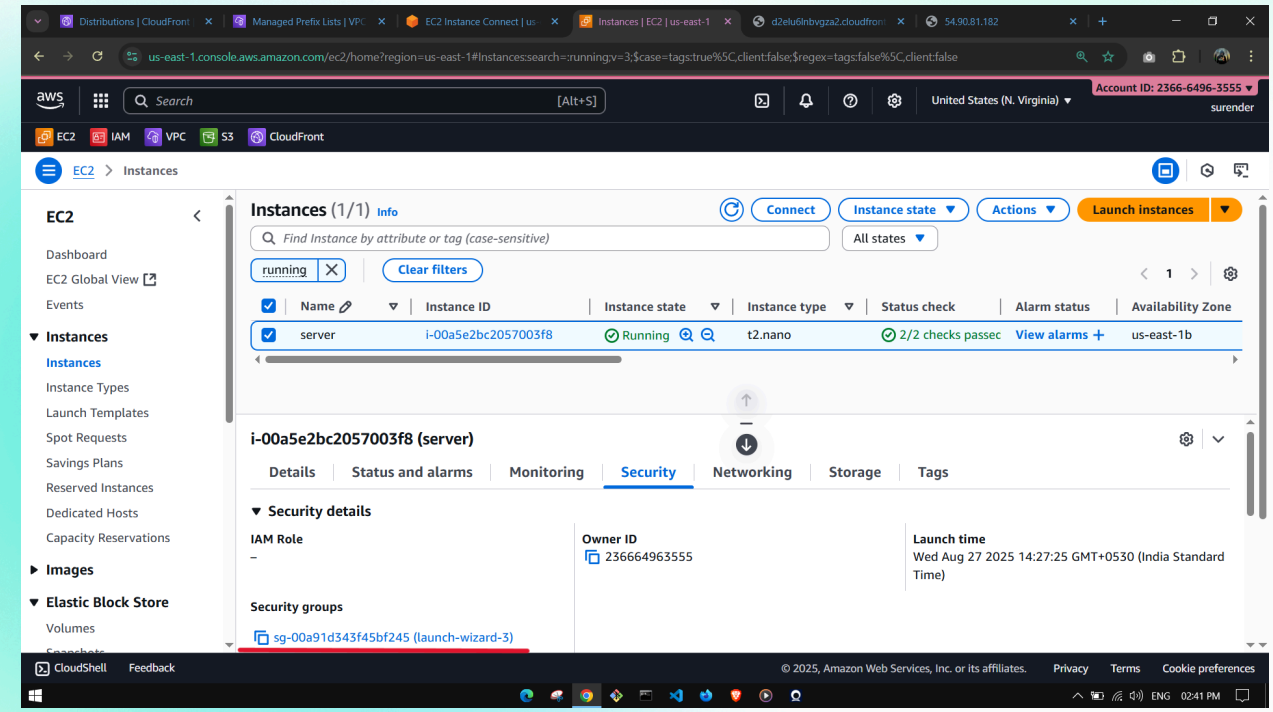
But there is a problem: it is also accessible through your public ip address of your server instance. It is not a good practice to be accessible through the server ip address so let's change it and make it secure.



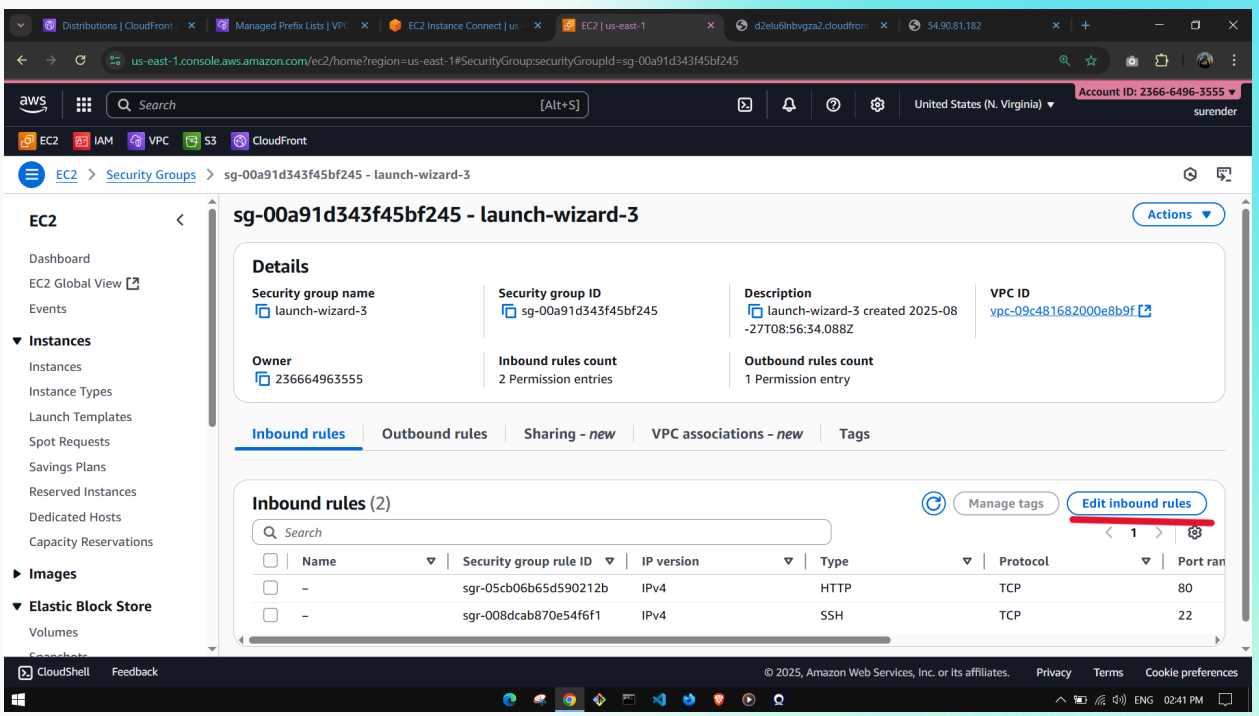
Now go to vpc service and copy the prefix-id of the cloudfront origin facing. It stores all the ip addresses of the edge locations of the cloudfront and aws manages itself.



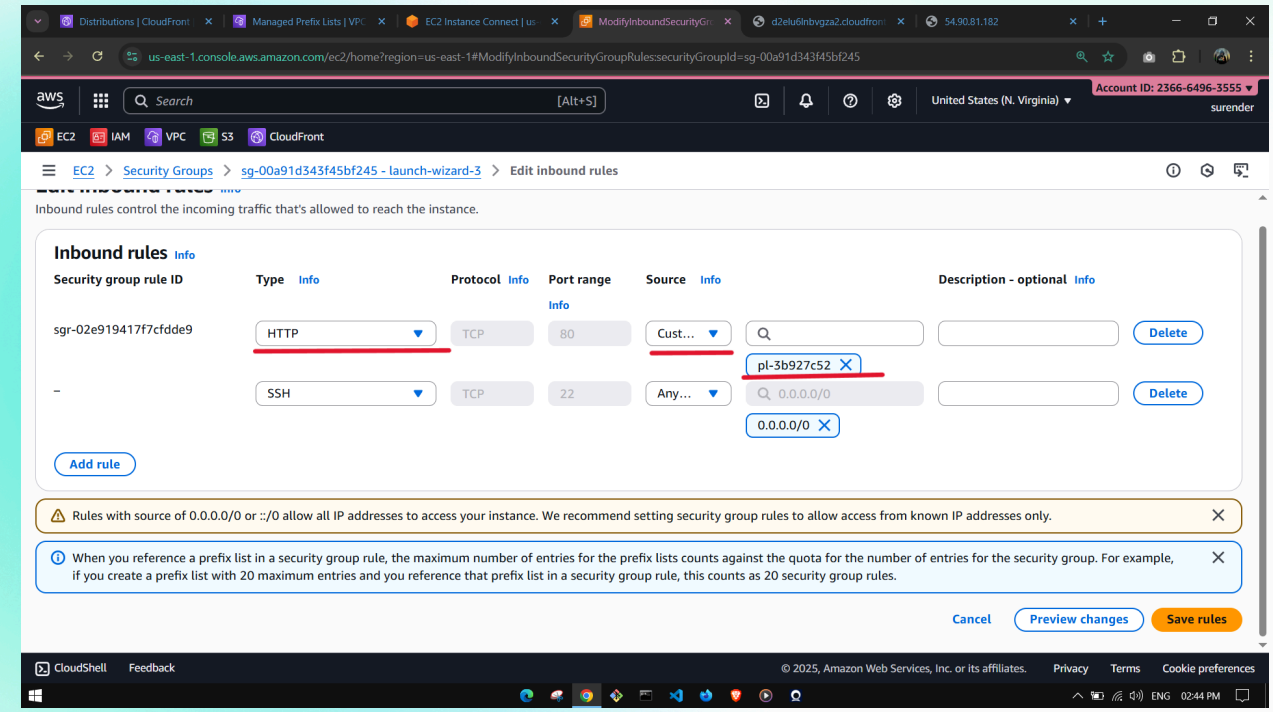
Go to the server security group and configure some settings in the inbound rule.



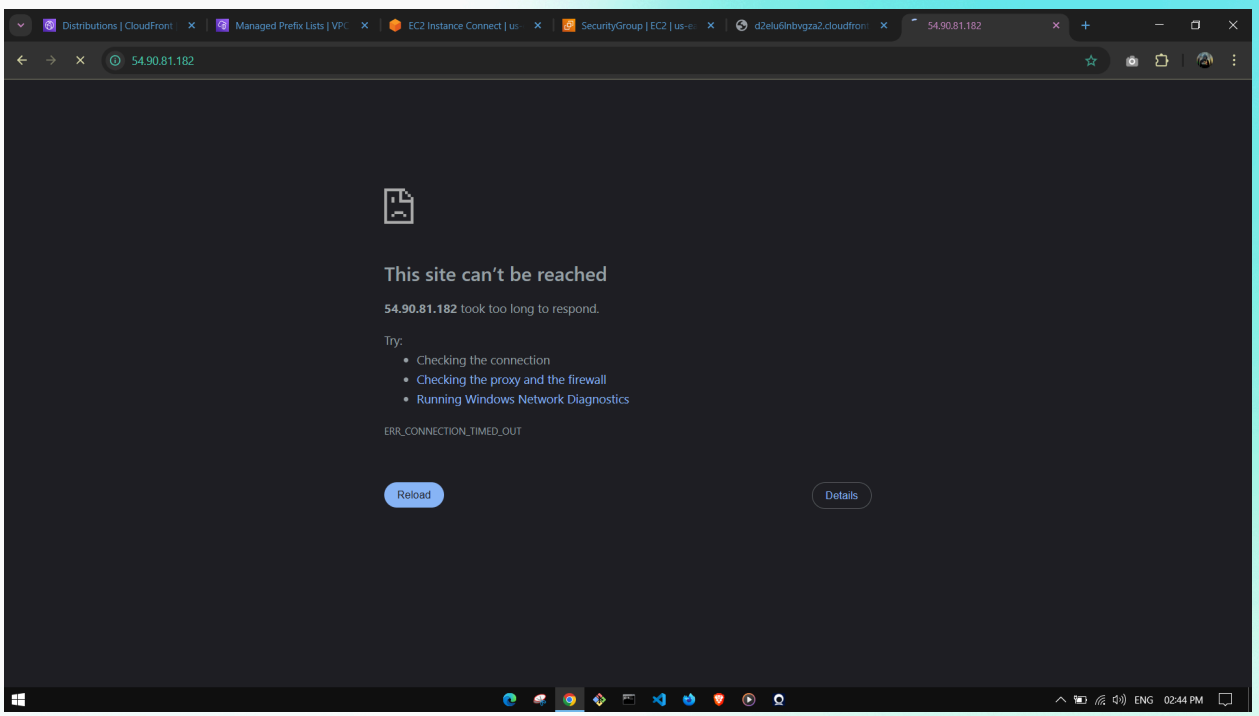
Click on edit inbound rules.



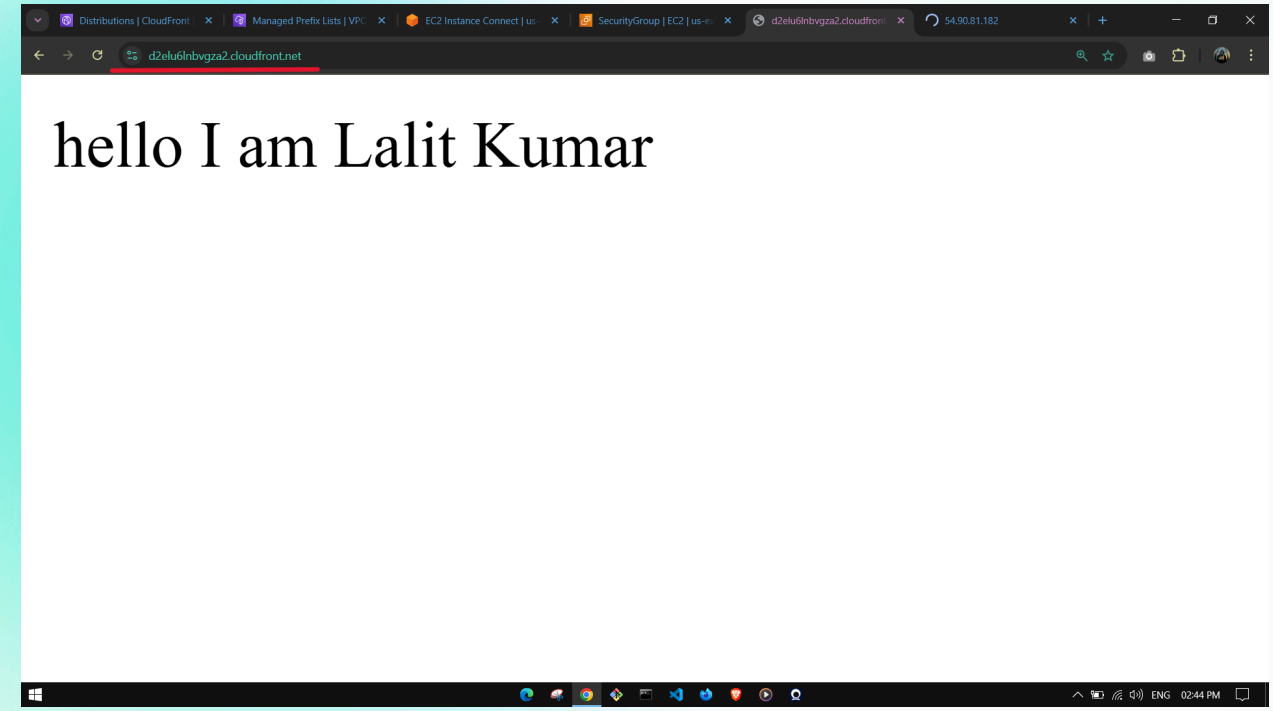
Remove your old http rule now add new and paste cloudfront prefix-id which you copied earlier in the source of http rule.



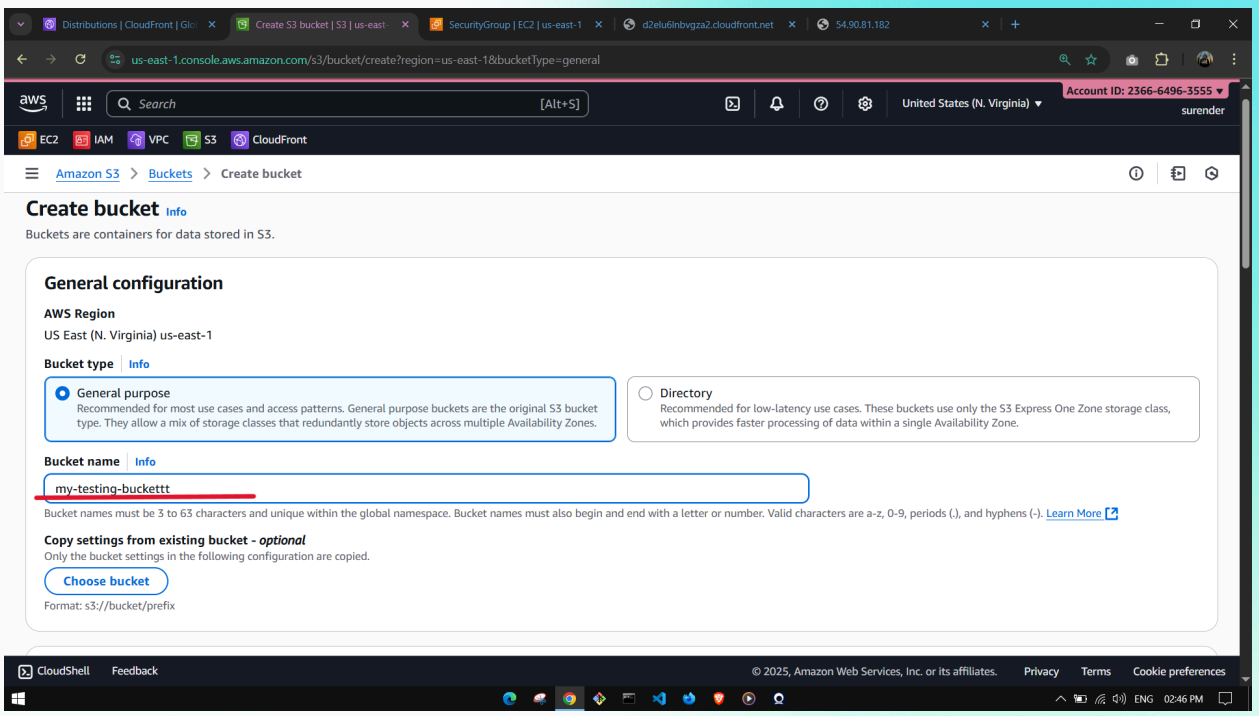
Now try to access your web-application through the ip address of your server instance, probably you will not be able to access it.

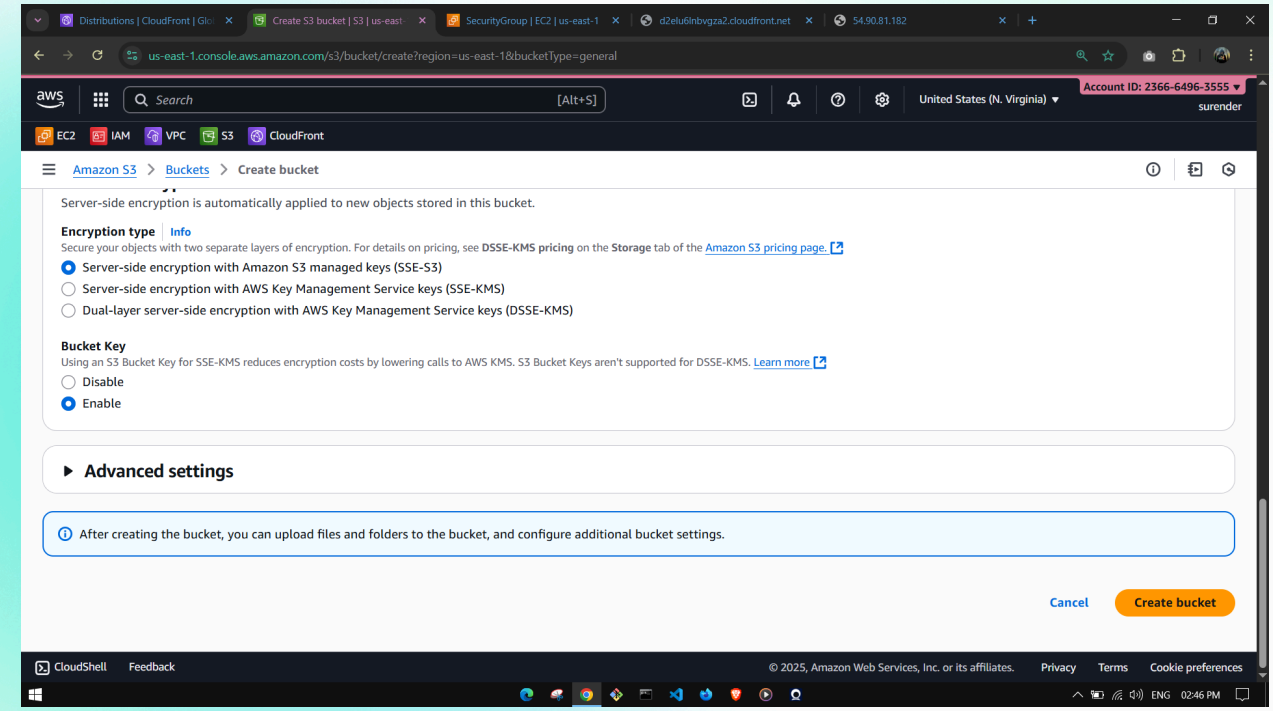


But it is accessible through your cloudfront



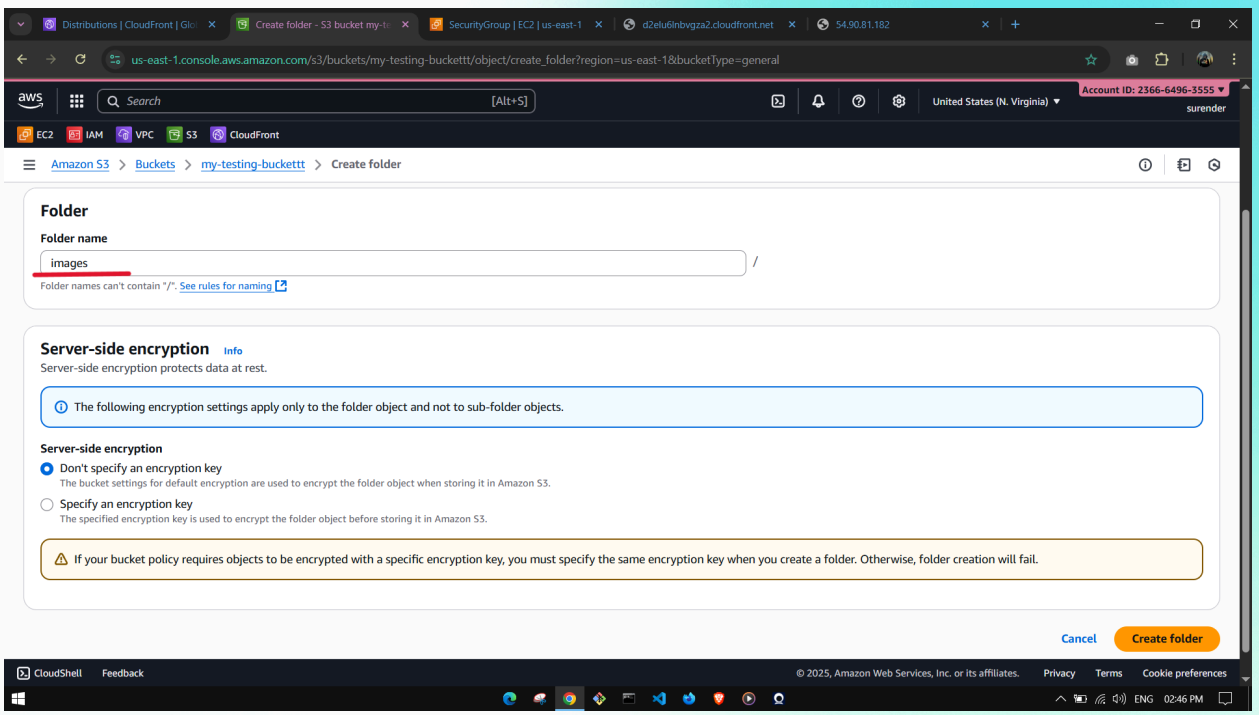
Now go to s3 service and create a s3 bucket for multiple origin in cloudfront



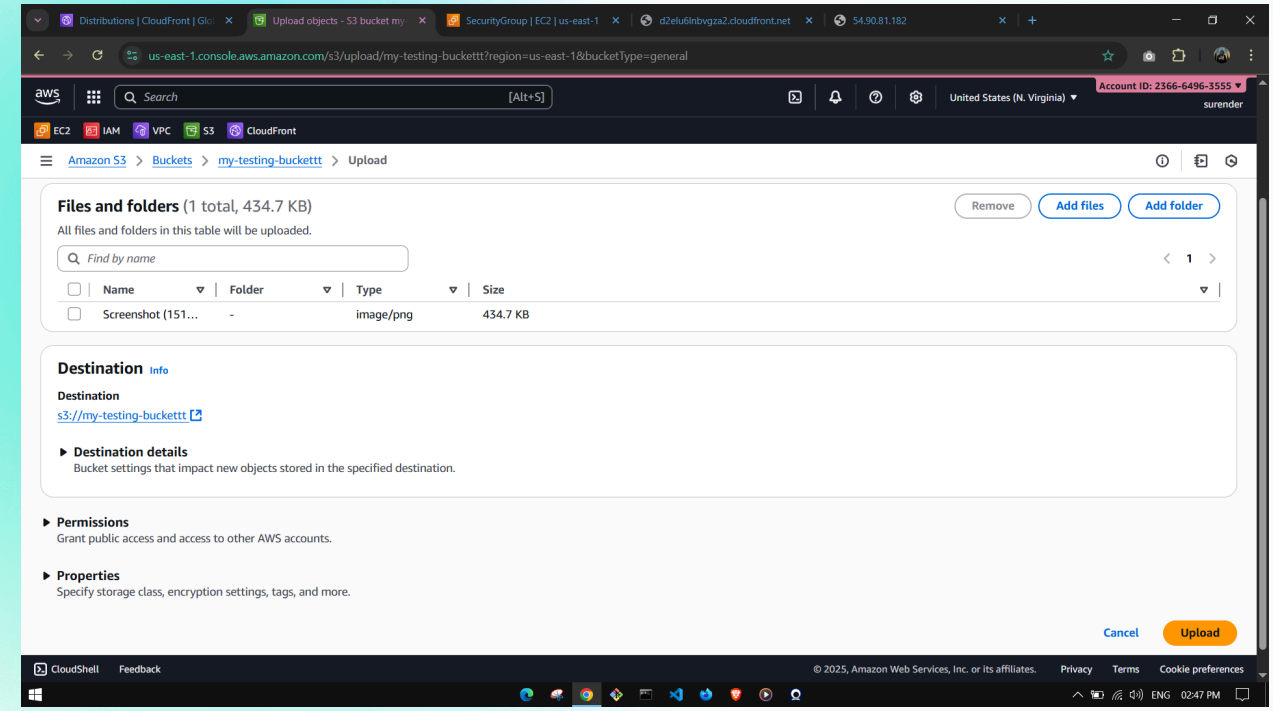


After creating the bucket just create a folder inside your bucket

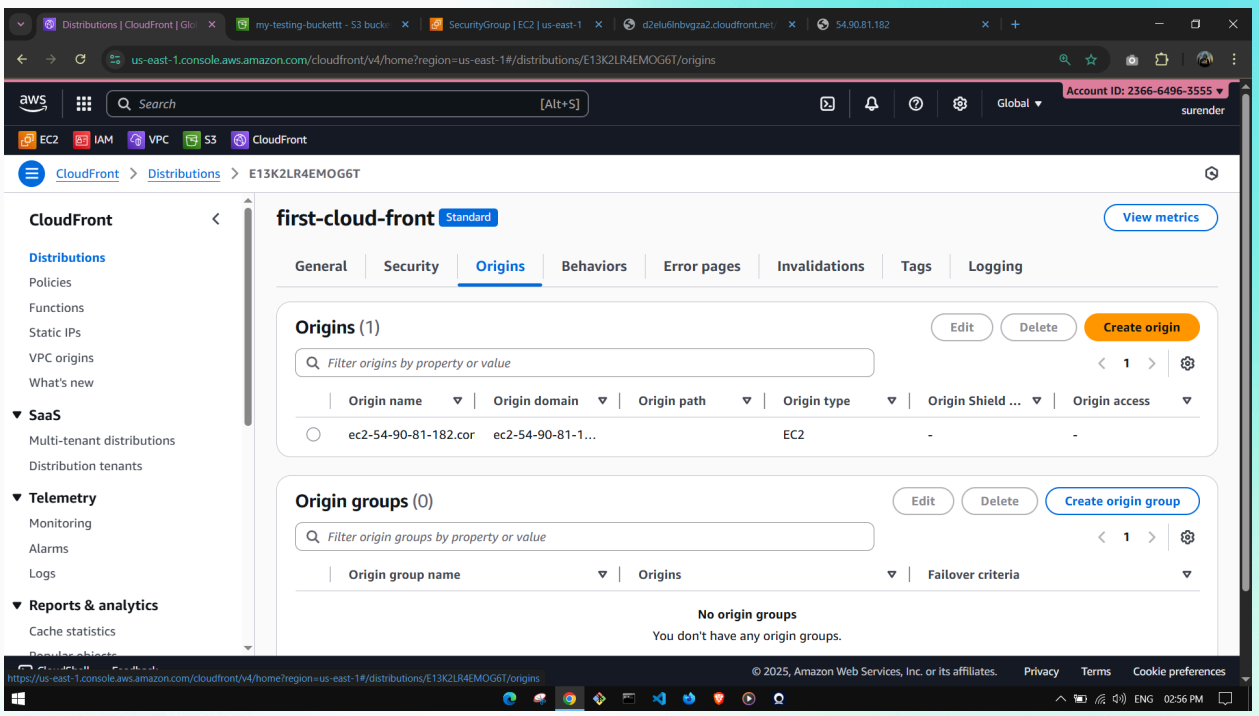
Create a folder with the name images as shown in the below image



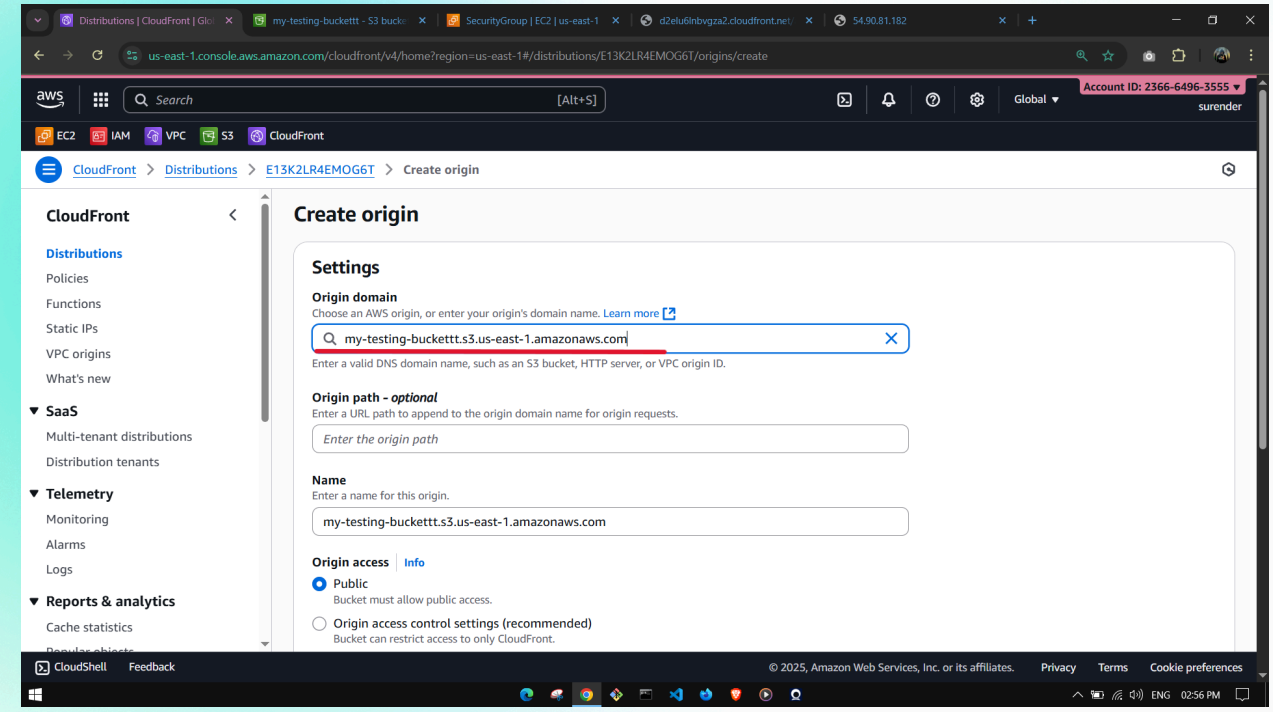
After creating the folder inside the bucket upload an image to the images folder of s3 bucket



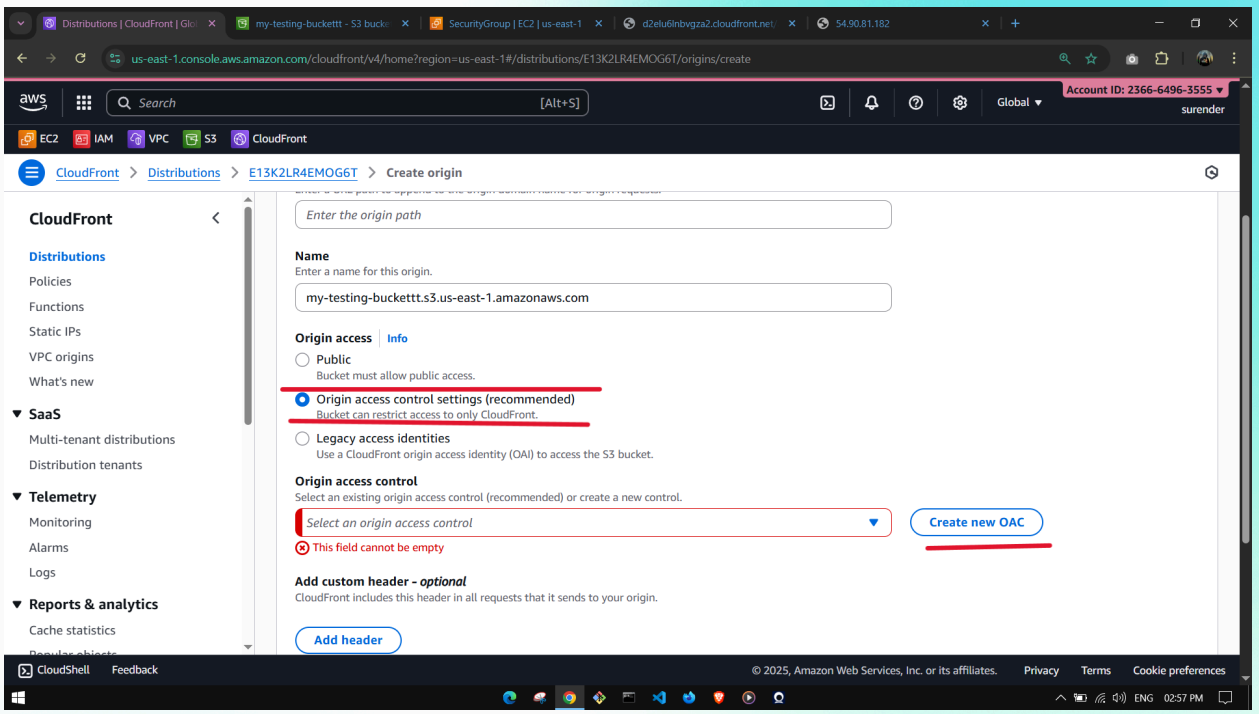
Go to the origin tab and see you have only one origin which is your ec2 instance (server).



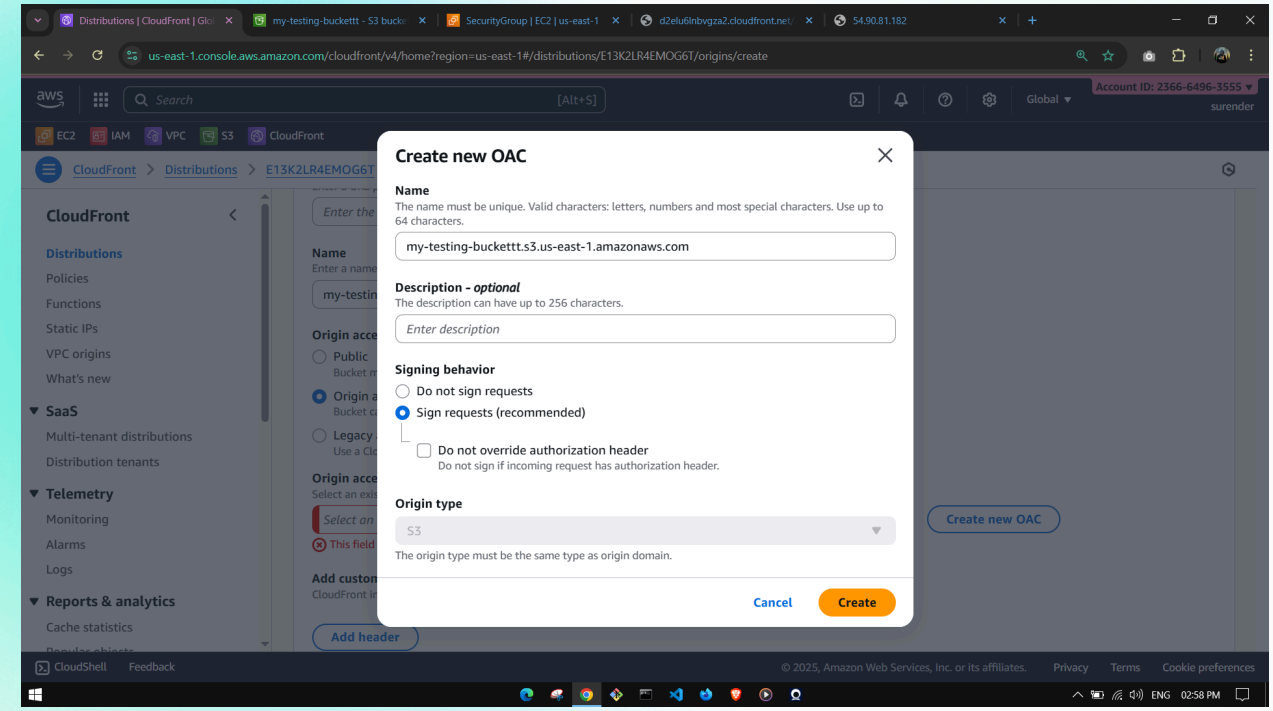
Create another origin for your s3 bucket and paste DNS of your bucket here



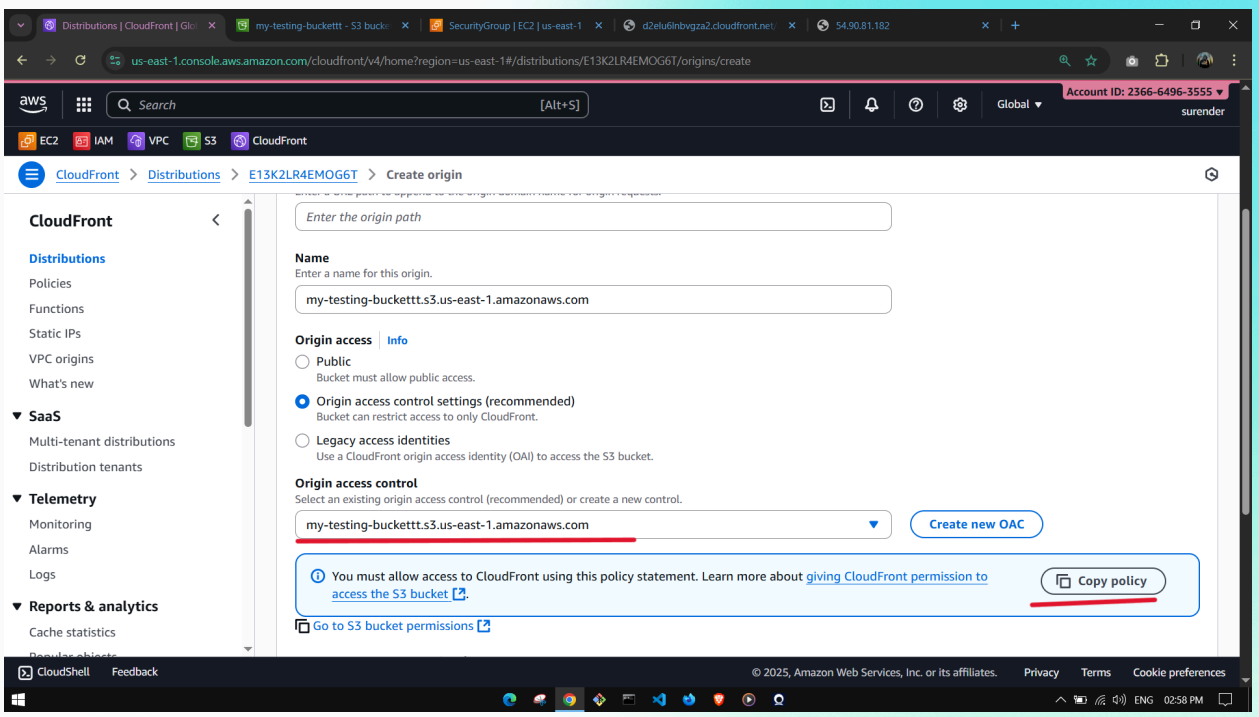
You don't have need to make your bucket public, select the setting which are highlighted in the below image and create **origin access control** to allow access the s3 objects.



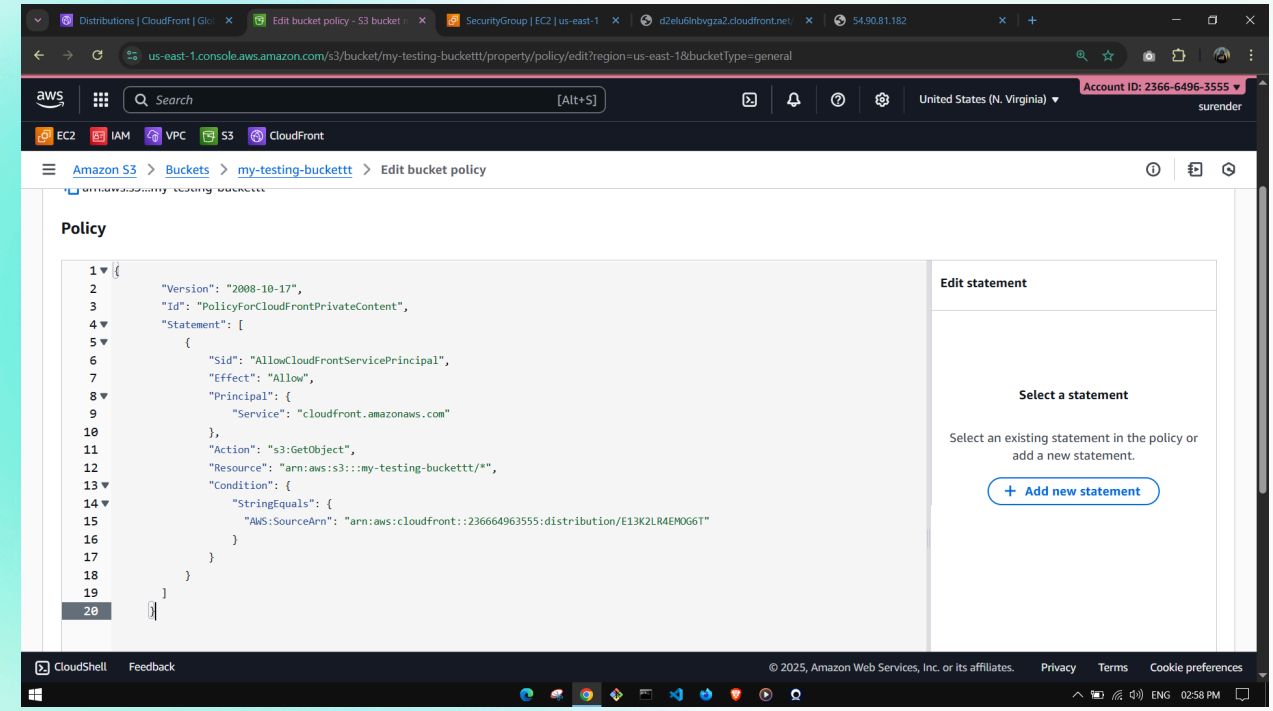
Just create an OAC or you can use what you created before.



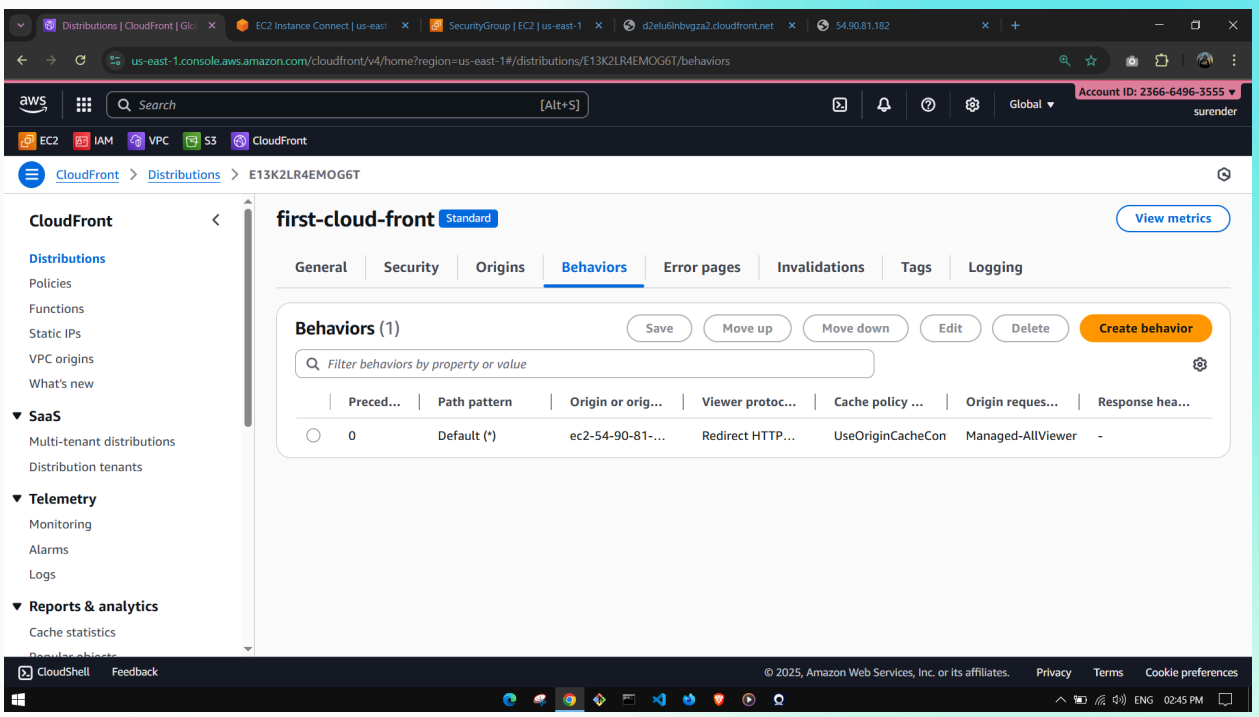
Select your OAC which you created right now and copy the policy to be paste in the s3 bucket policy later.



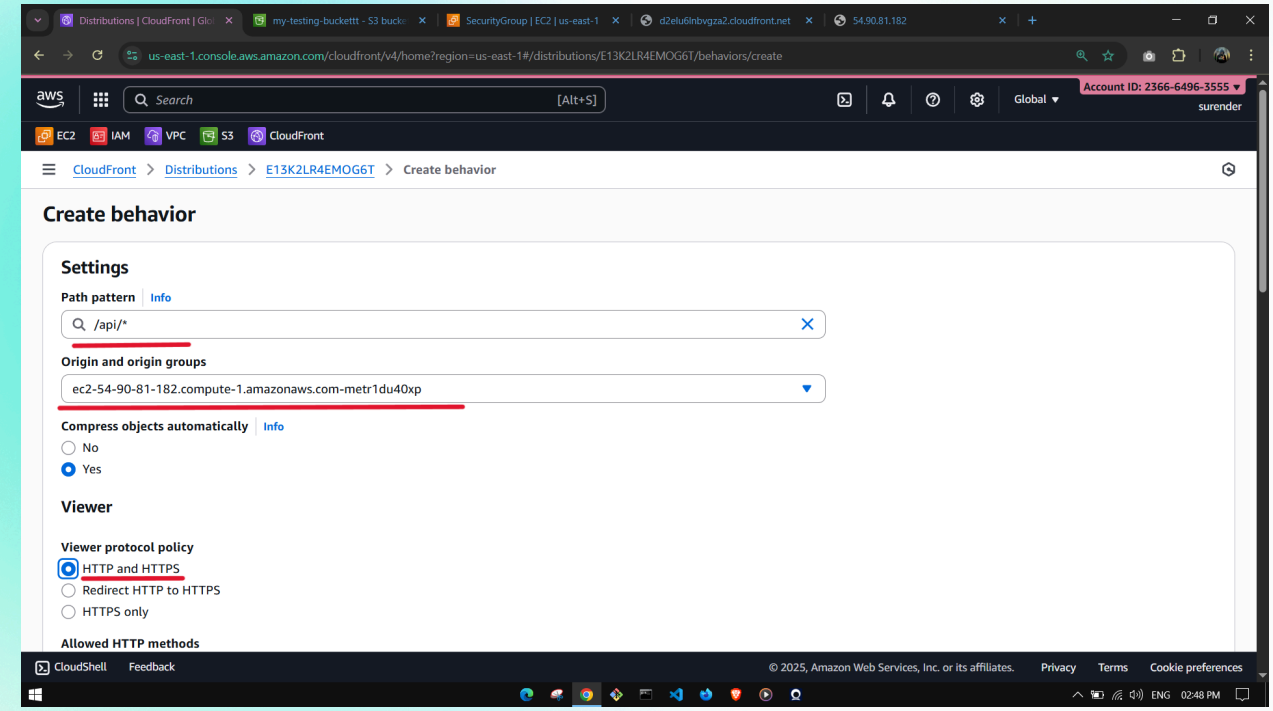
Paste the policy which you copy earlier to the s3 bucket policy.

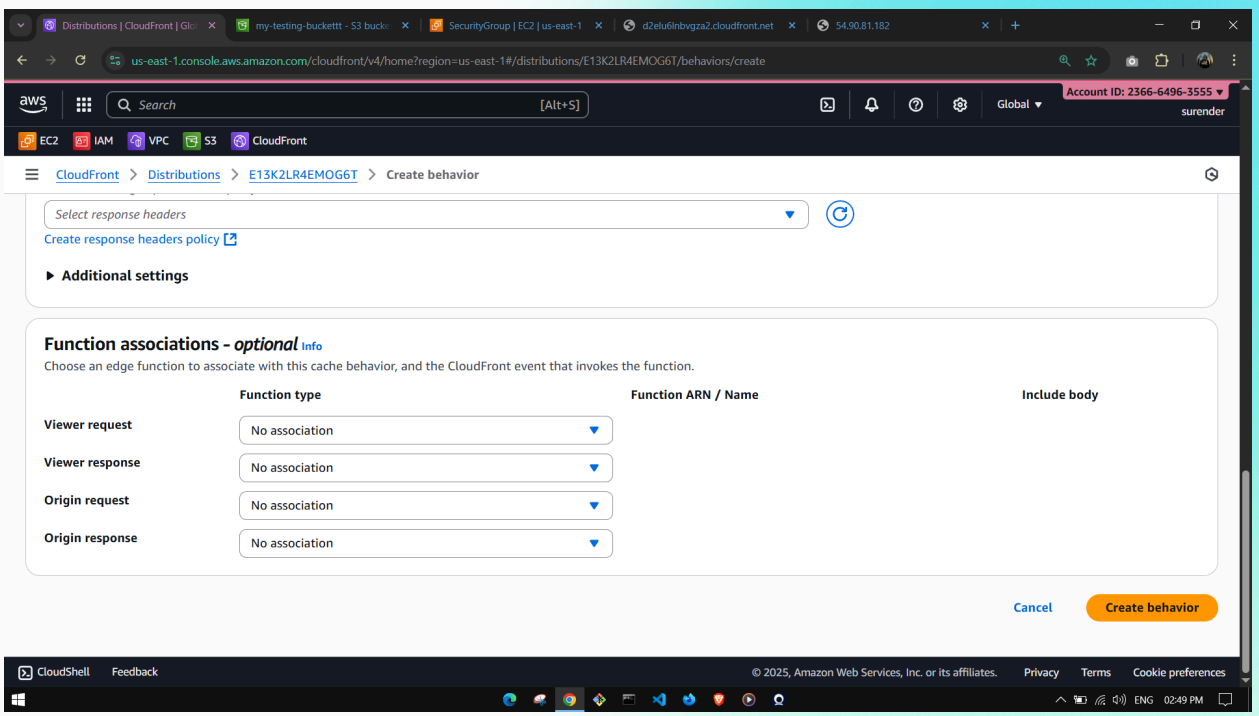


Go to the behaviour tab of your cloudfront and create another behaviour for your application or you want to redirect your traffic for a specific path.

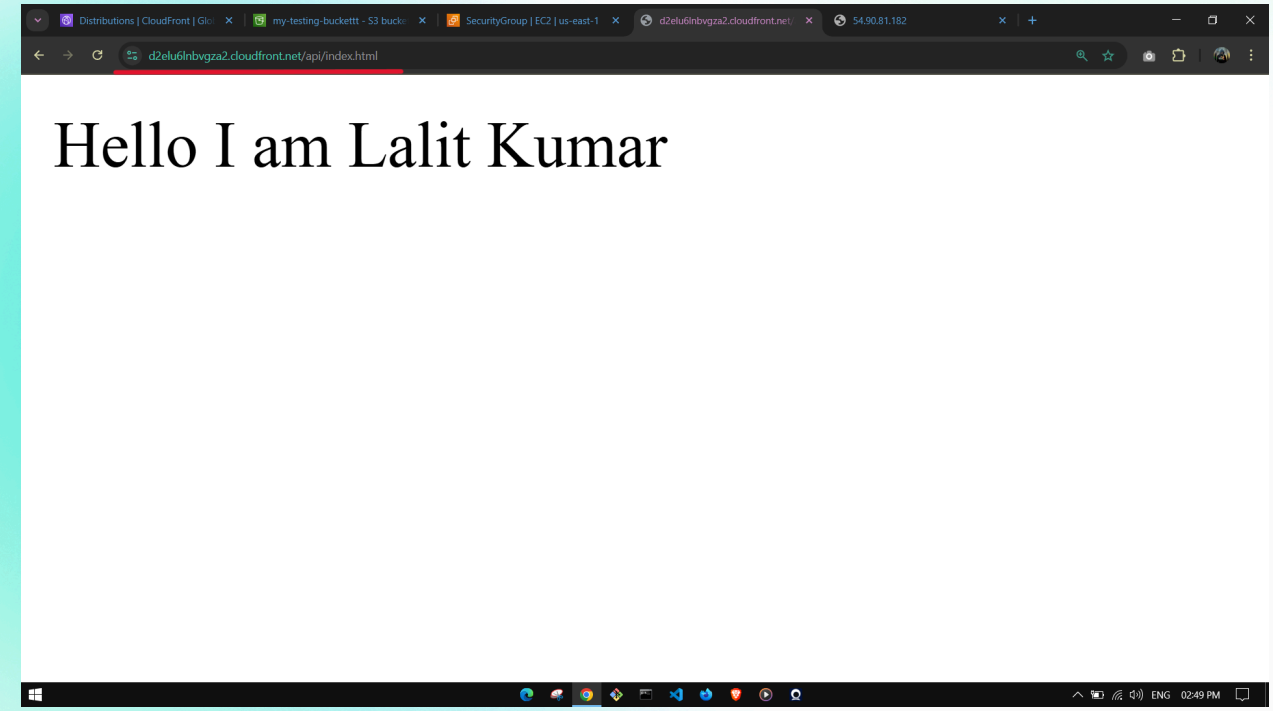


Create a path and select origin as your ec2 instance and create a behaviour for this path.

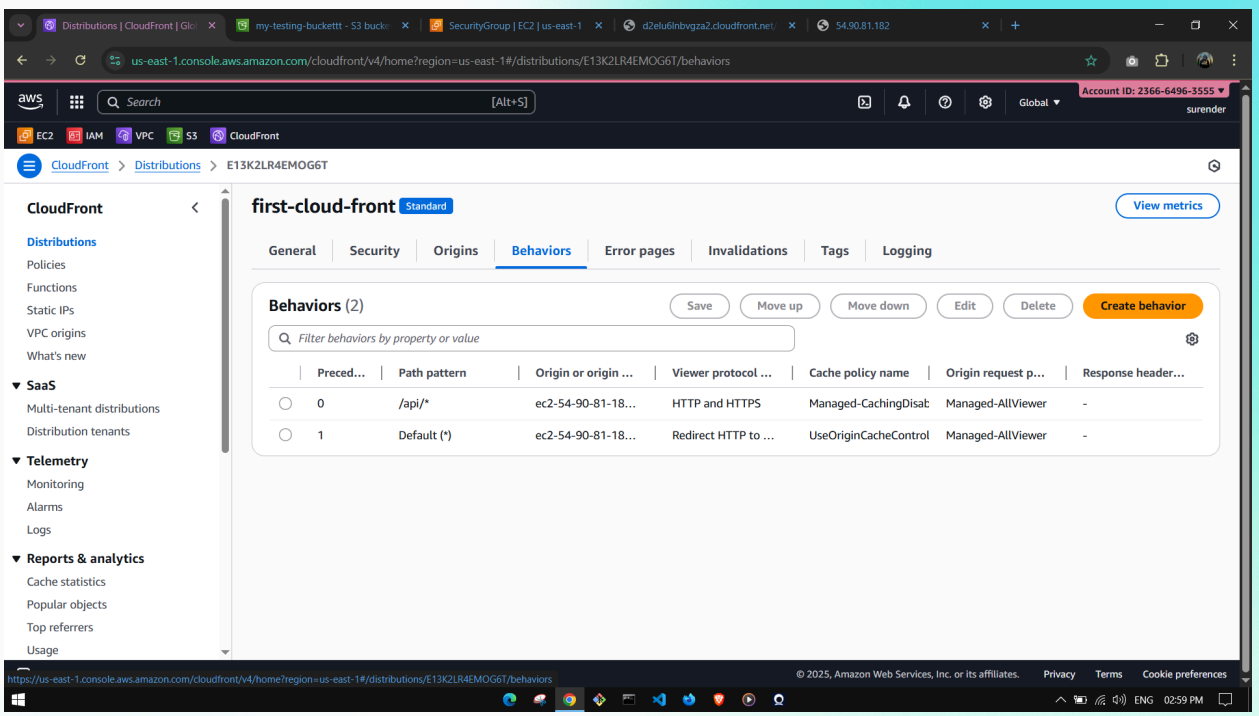




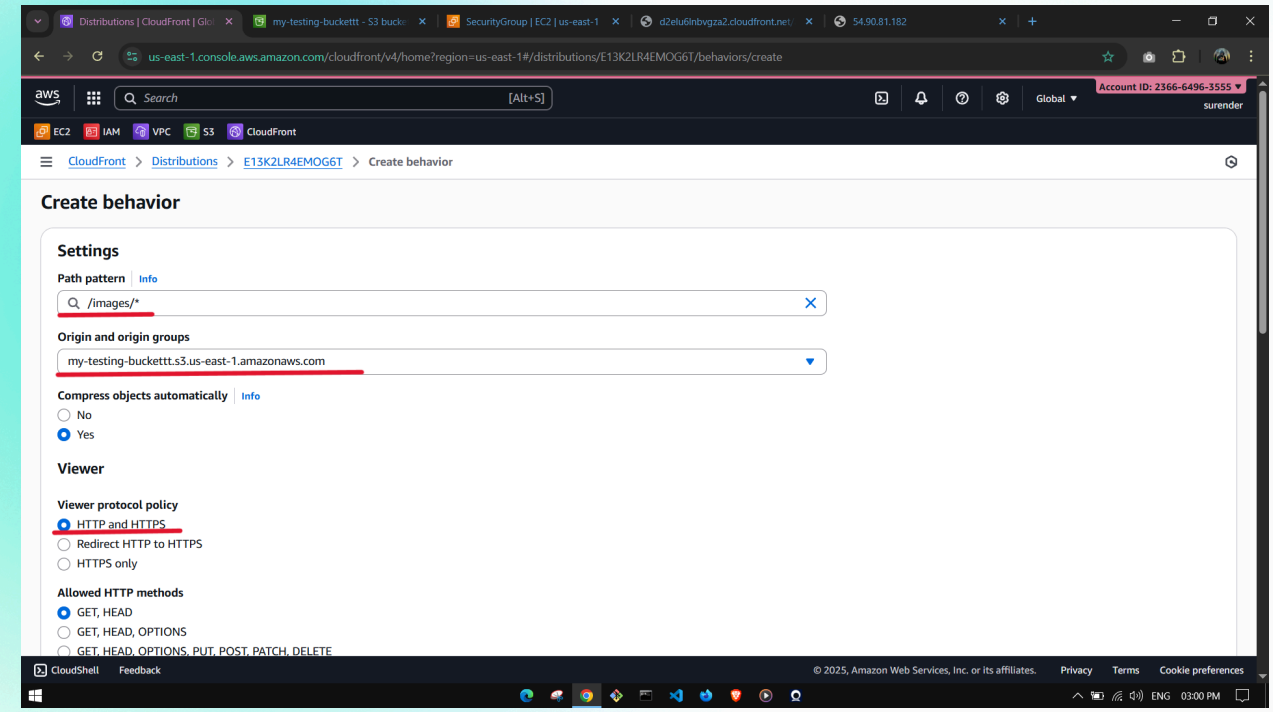
Check the path of your cloudfront with **/api** and you will see your behaviour works successfully.



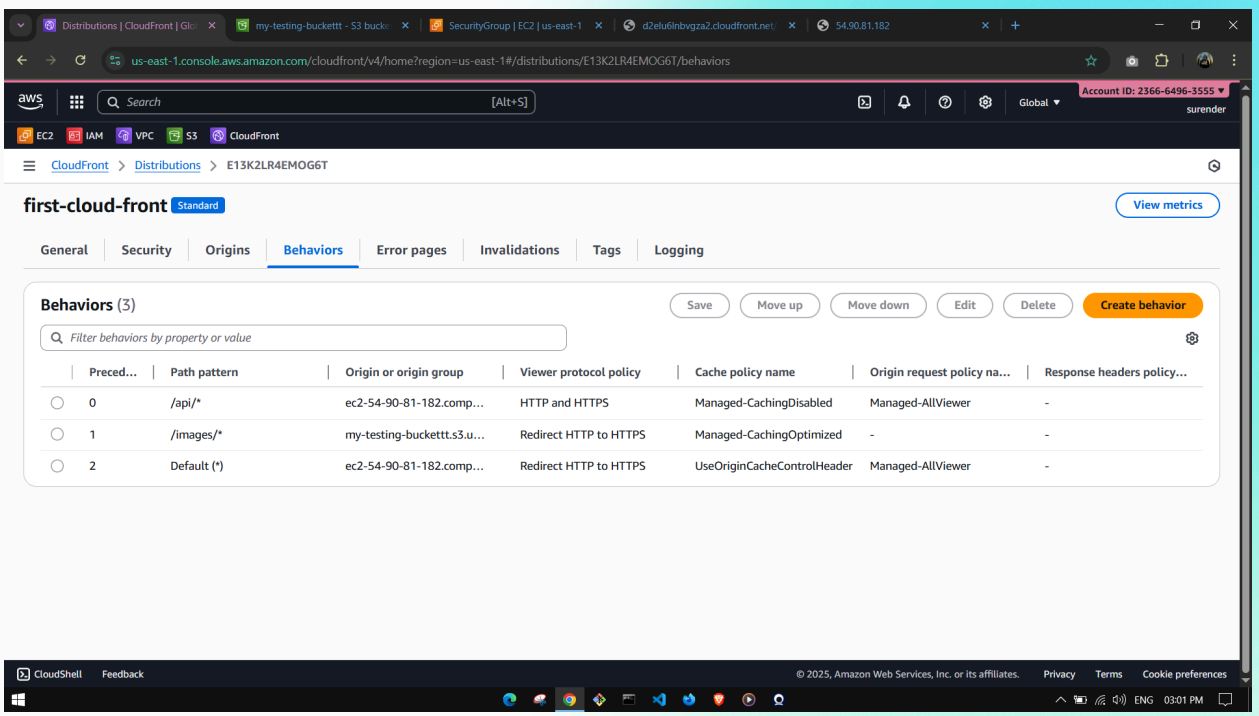
Go to the behaviour tab and you will see your have two behaviours default and **/apt/***



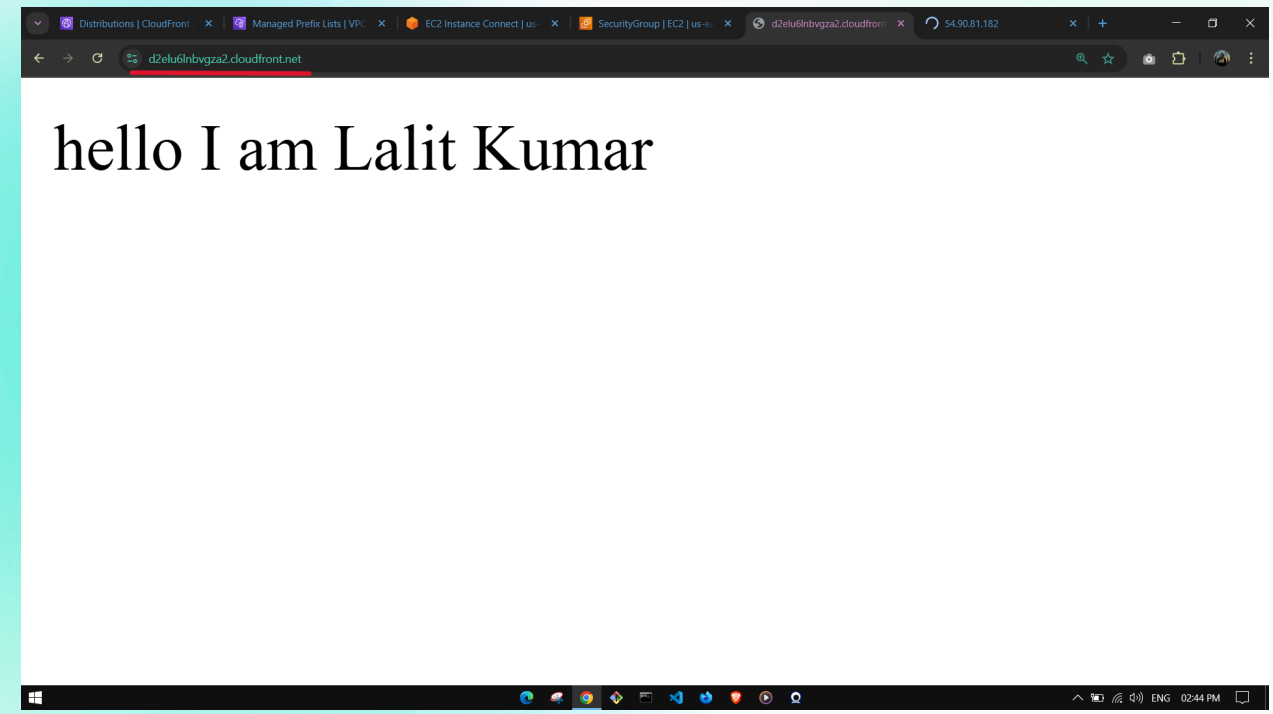
Create a path **/images/*** for redirect to your bucket and select your bucket as your origin also choose the settings which are shown in the below image and create it.



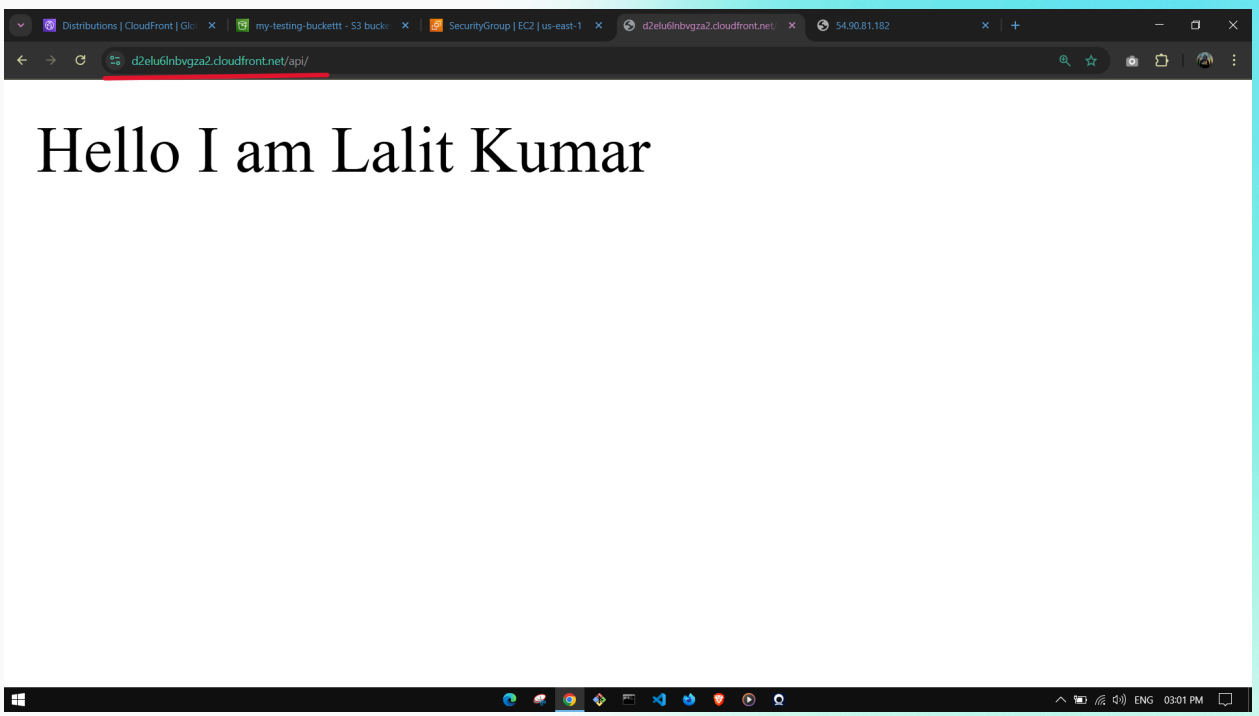
Now you can see clearly there are three behaviours and let's test them.



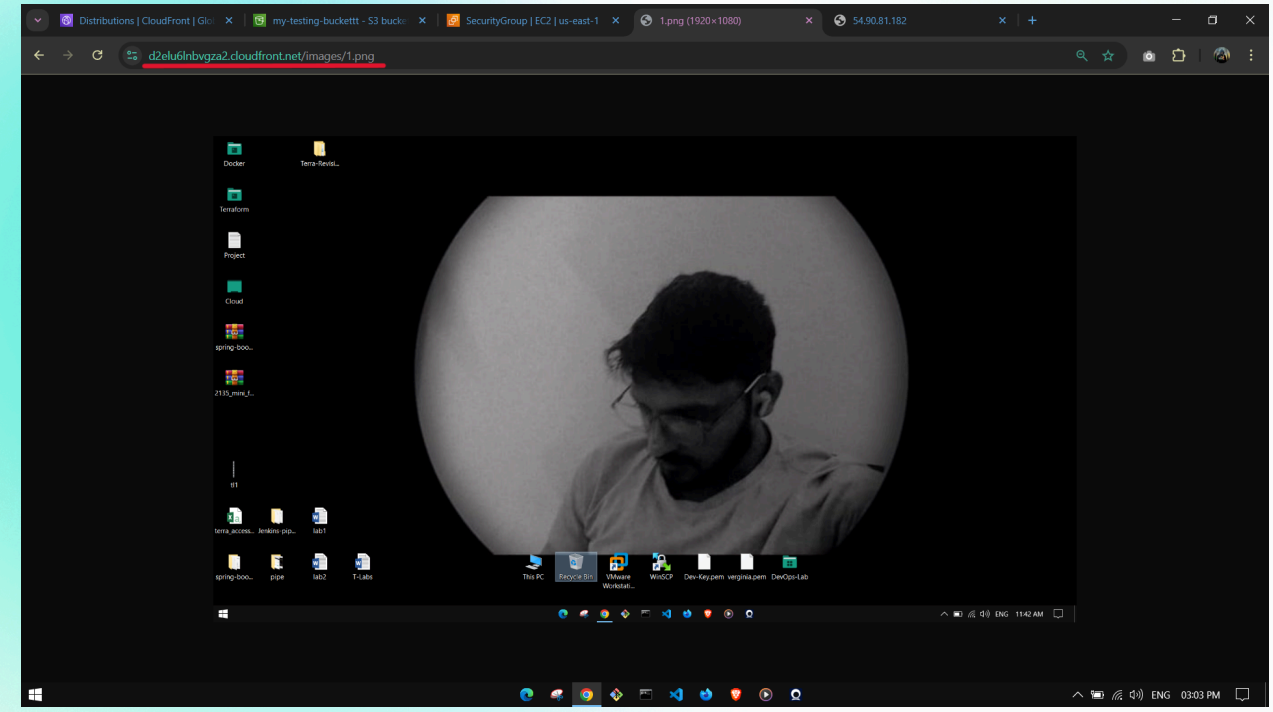
Default one



Behaviour **/api/***



Behaviour **/images/*** you will have the images age accessible through the cloudfront even if the bucket is private. Even you can use bucket for other purposes like static website hosting, etc



Let's Connect :-

Lalit Kumar

Follow: www.linkedin.com/in/lalit192977

Profile: https://bento.me/lalit192977