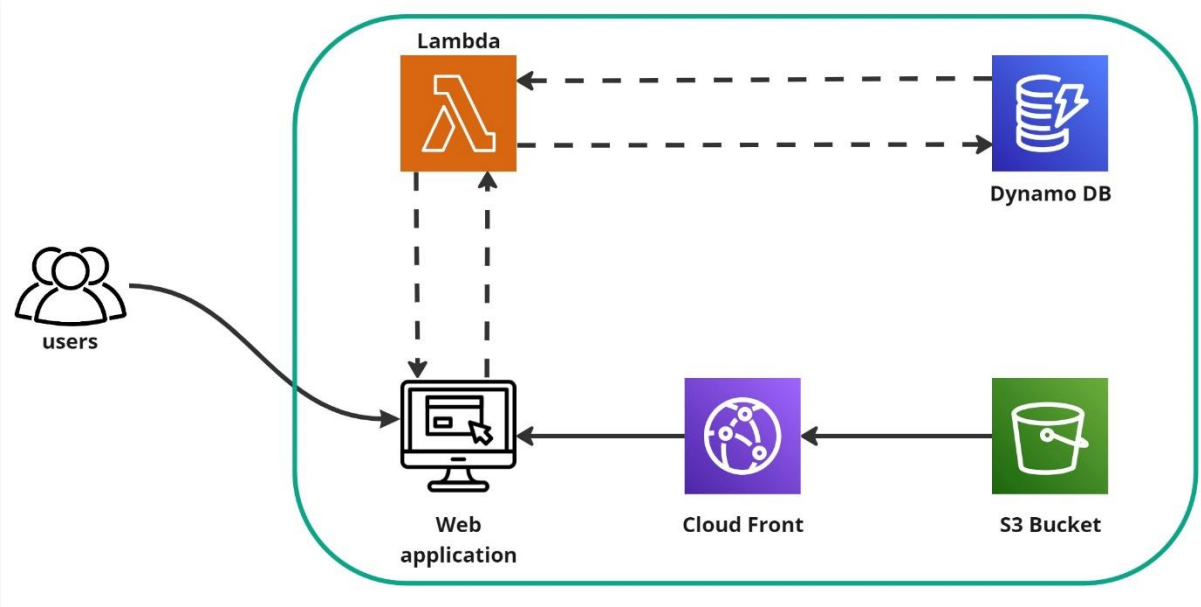
Serverless web application

Project Description:

In this project, you will build a serverless web application using AWS Lambda, CloudFront, DynamoDB, and S3. The application will allow users to update the view count from a DynamoDB table.

Project Architecture:



Steps to build the project:

- Create a DynamoDB table to store the items as view count.
- Build a Lambda function to handle the view count operations on the DynamoDB table.
- Use S3 to store and host the web application's static files (HTML, CSS, and JavaScript).
- Create a CloudFront distribution to serve the S3-hosted static files with low latency.

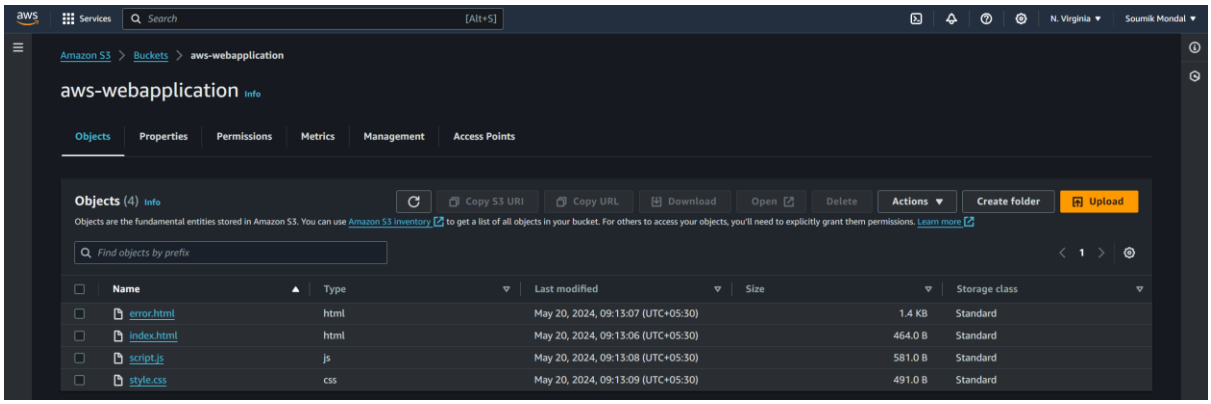
Expected outcome:

Upon completing the project, you will have a working serverless web application hosted on AWS. Every time you refresh the webpage, the view count will increase.

This project will help you improve your skills in cloud computing, serverless architecture, and AWS services.

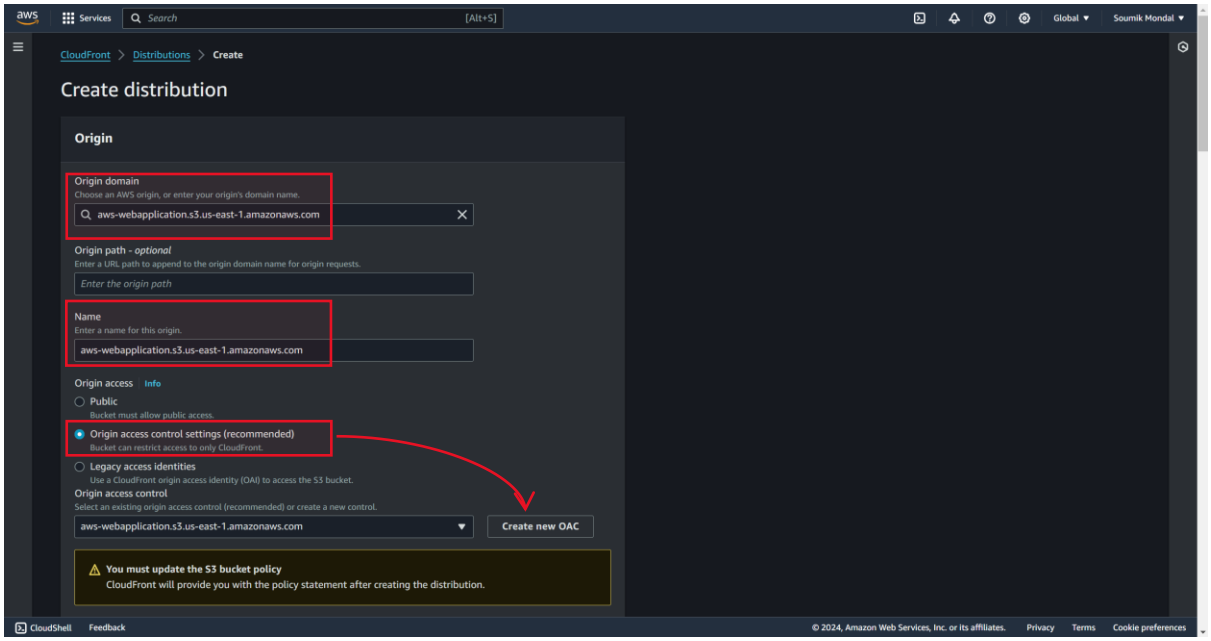
Code repo: https://github.com/MiksVeg/aws-webapplication.git

Let’s create a S3 bucket-

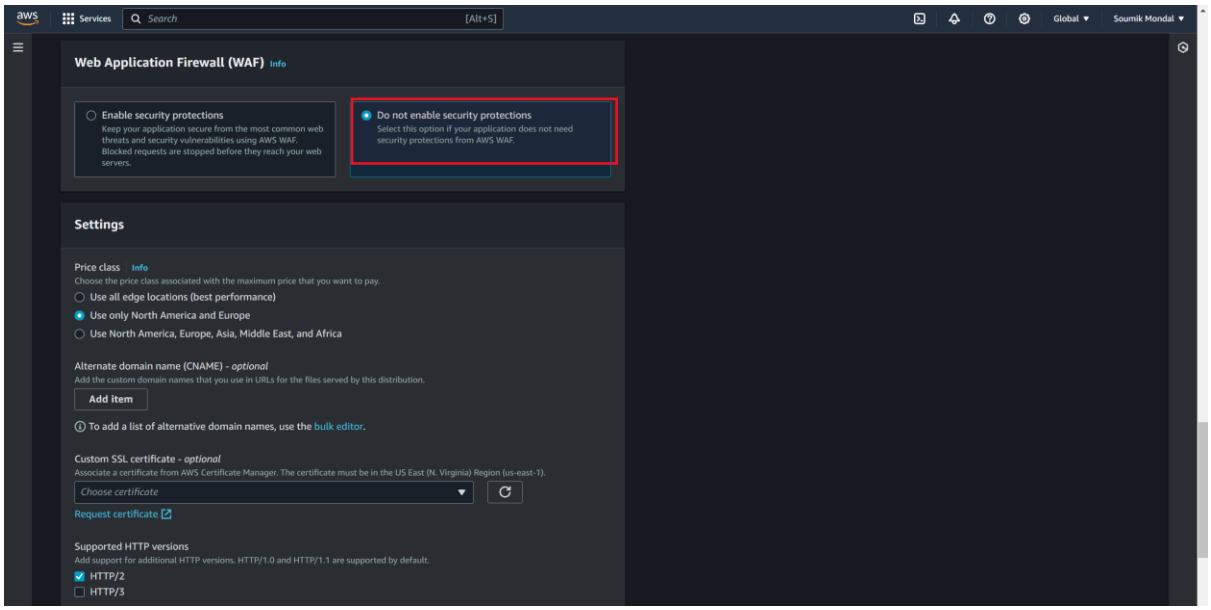


Created an S3 bucket and uploaded index.html, script.js, and style.css files

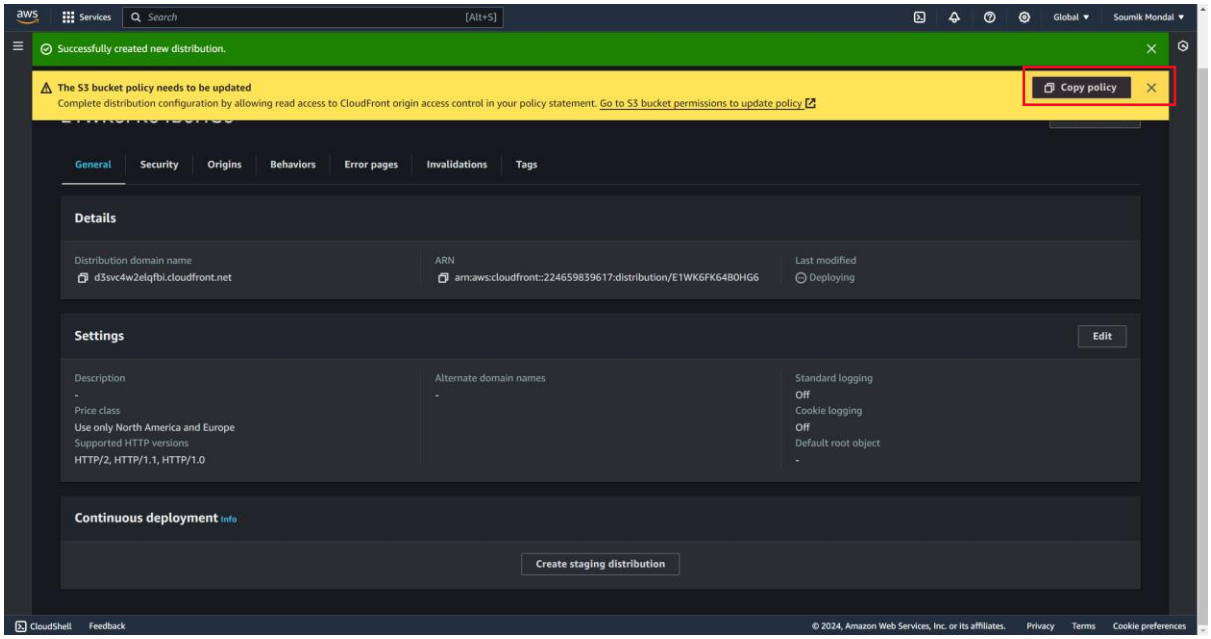
Now, create cloud front distribution -



Create a new OAC (Origin Access Identity) so that only CloudFront can access S3 bucket

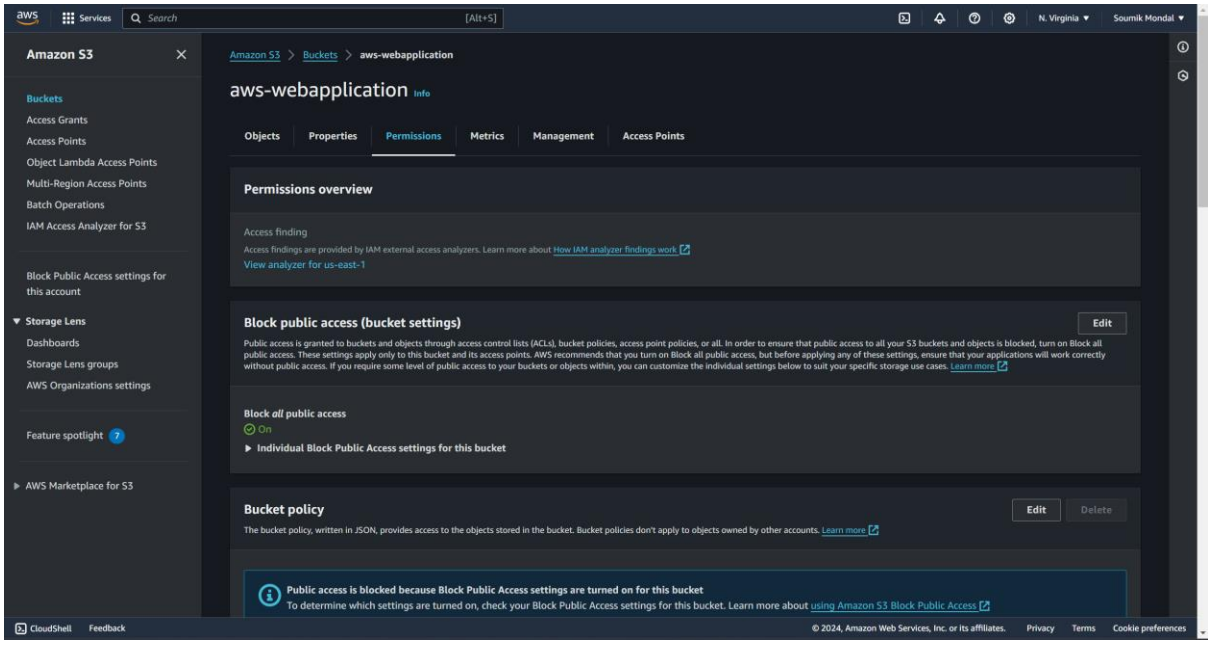


Now, we have to give permission to CloudFront to access the files of the S3 bucket -

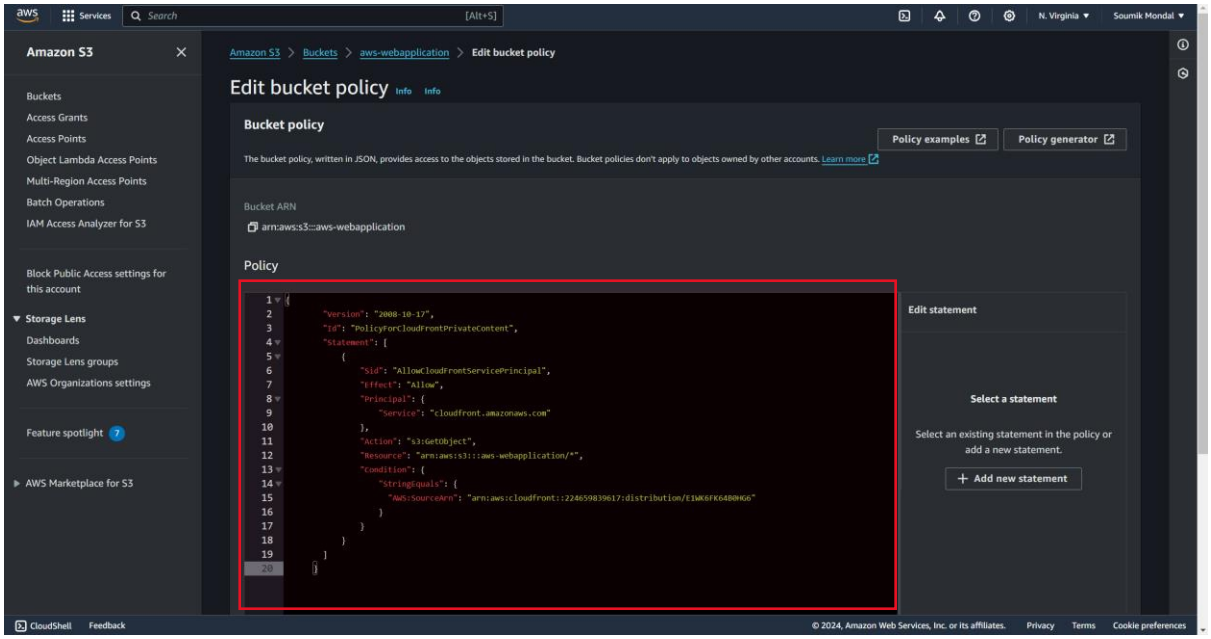


Copy the policy

Paste the policy in the Bucket policy of S3 -

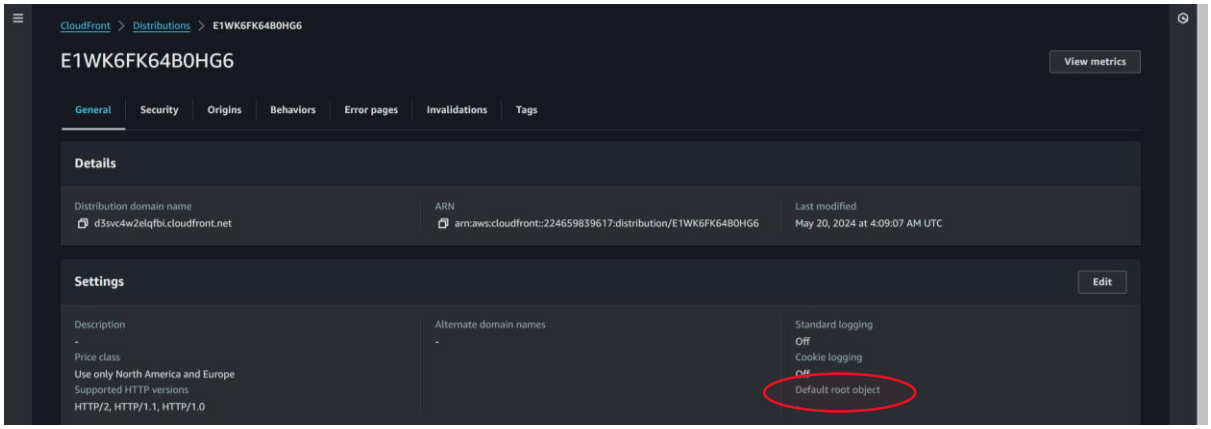


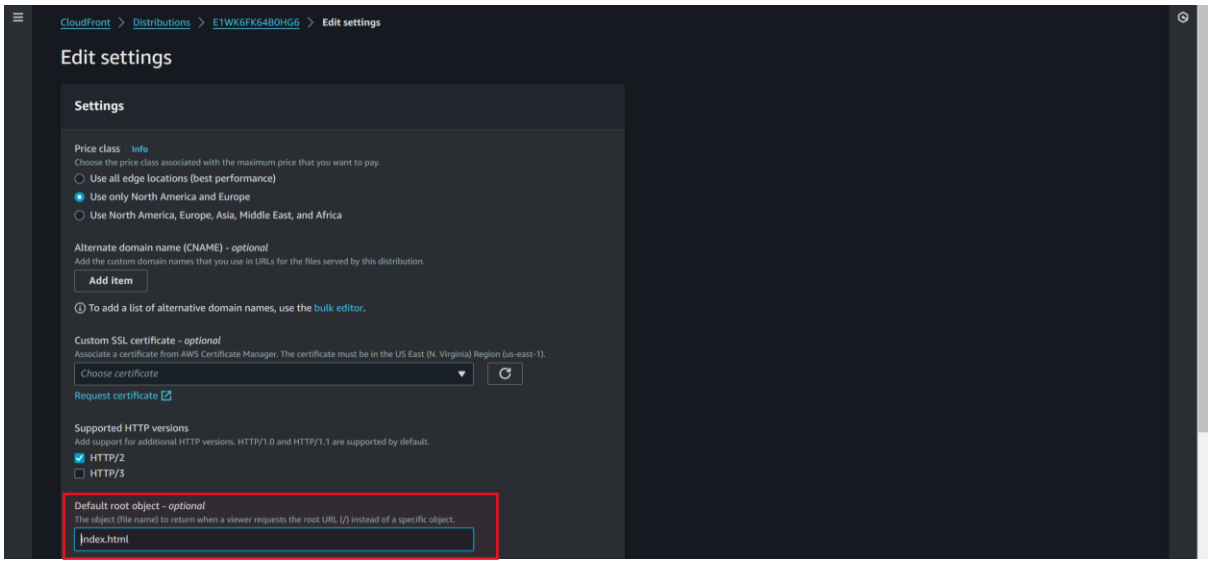
Go to the S3 Bucket and find the permission tab.



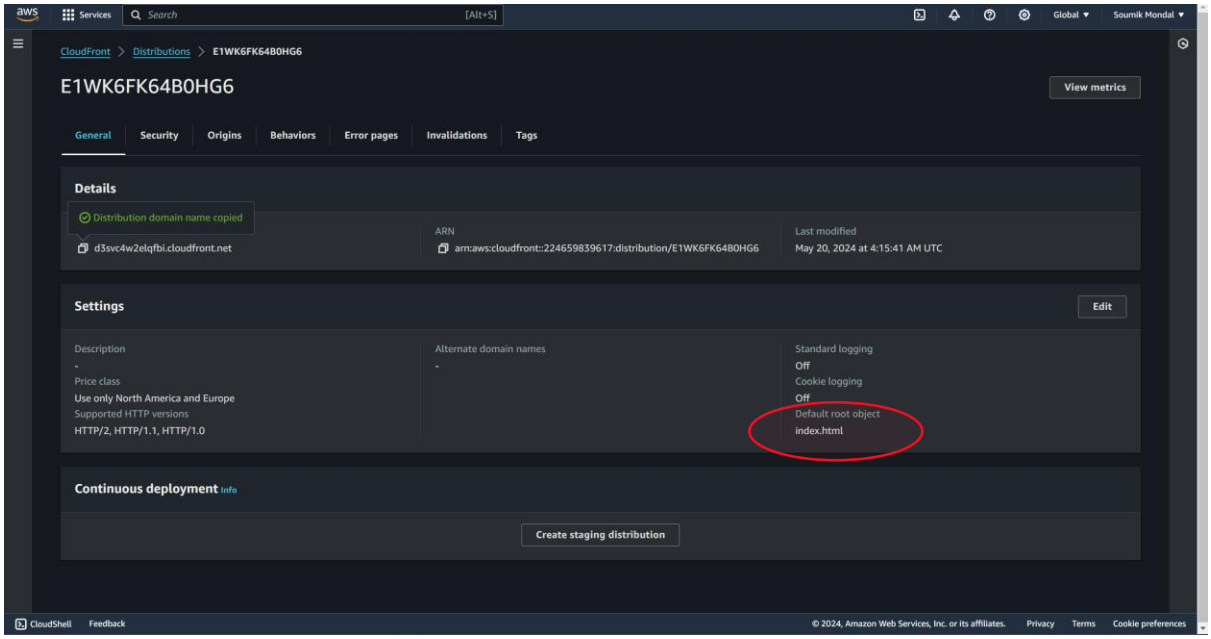
Paste and save the bucket policy

Now, configure the cloud front to face the home page to index.html -

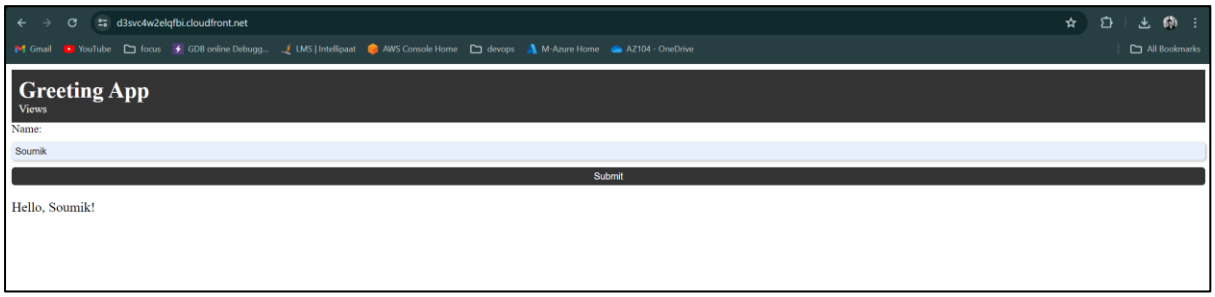




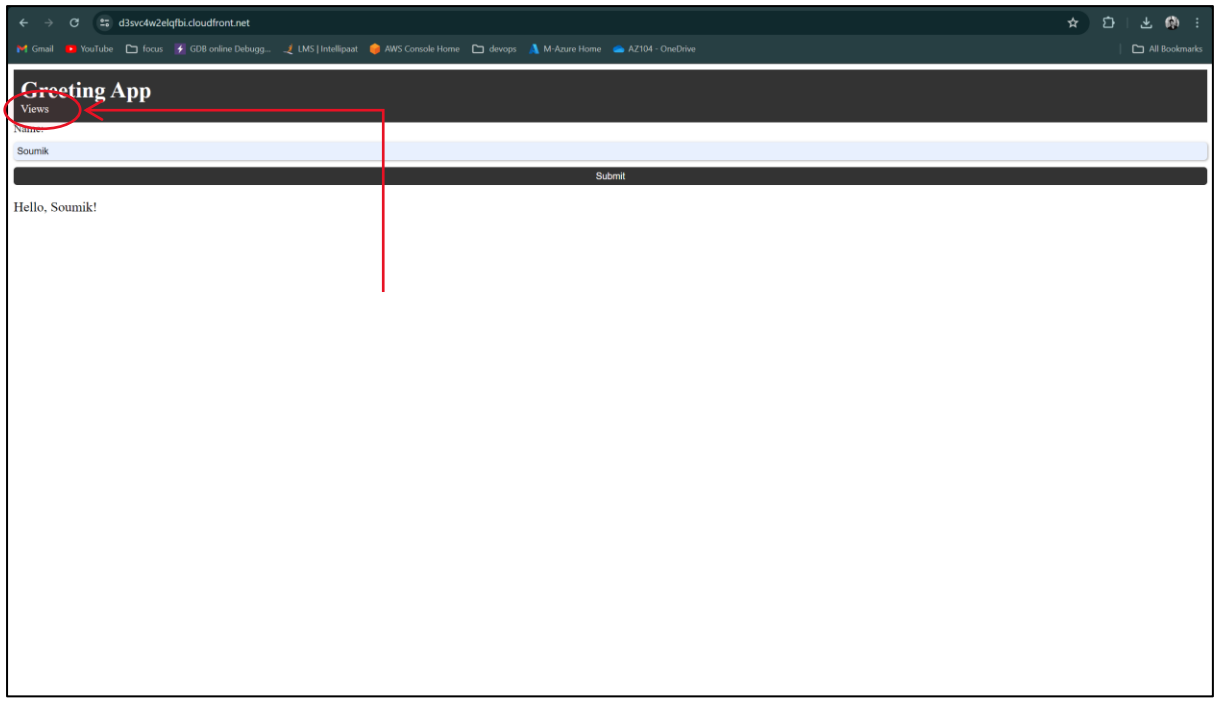
Edit the default root object to index.html



Successfully configured CloudFront

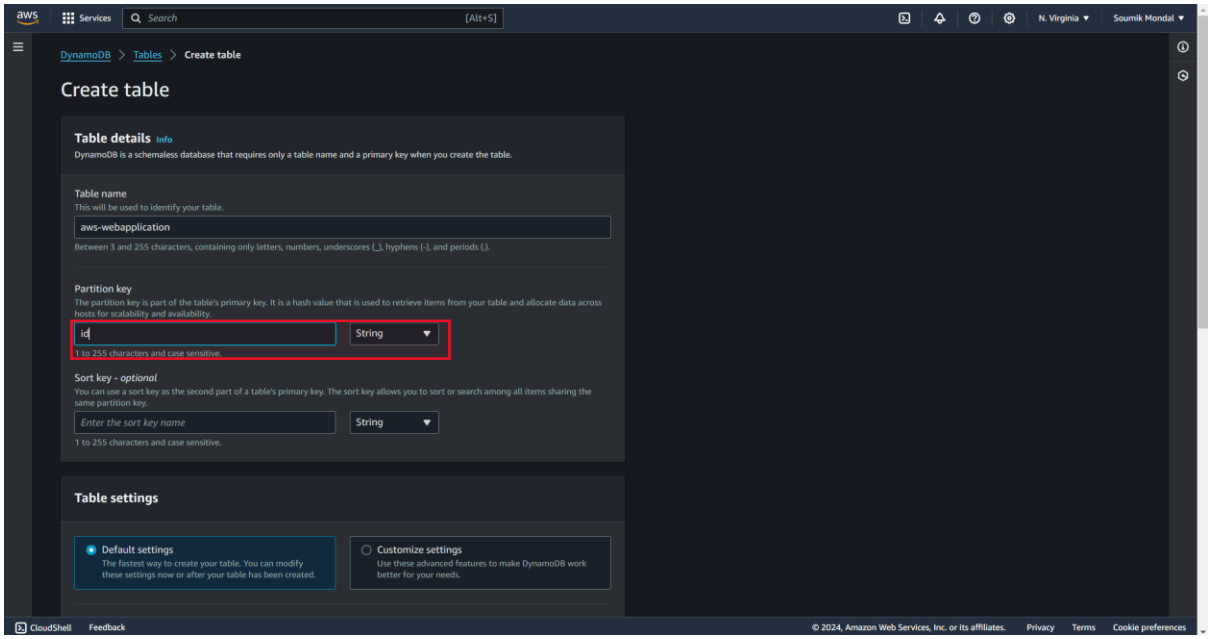


Successfully deployed and accessed the webpage.



Now, we will work on view count, which will increase when we refresh the page.

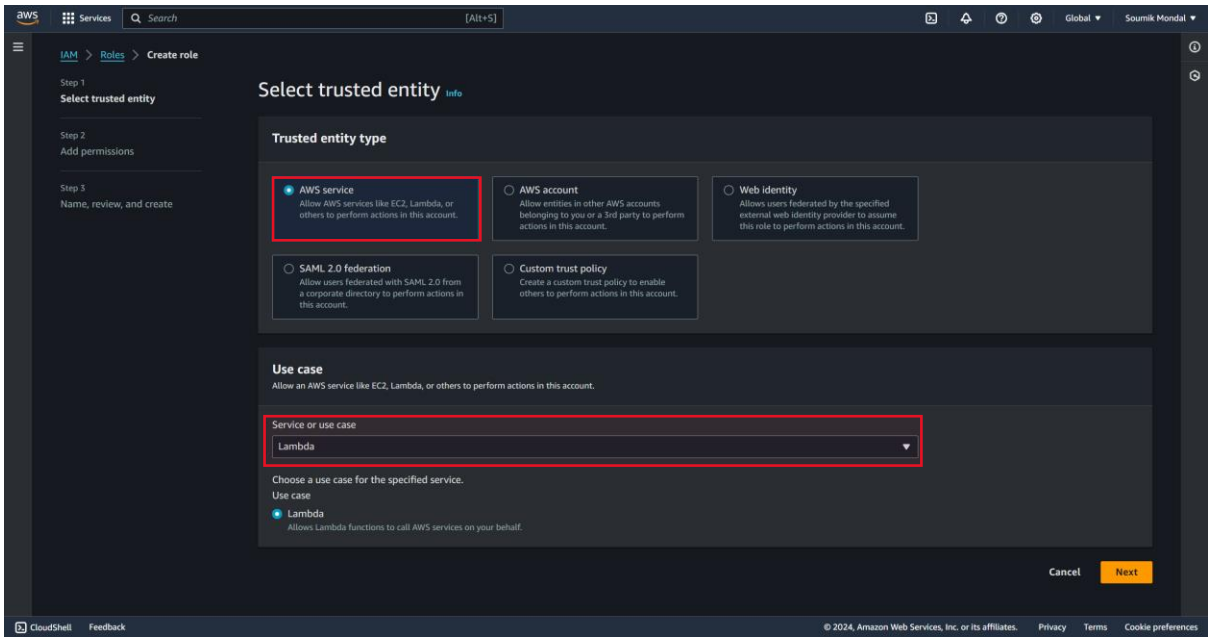
Let’s create a DynamoDB table -

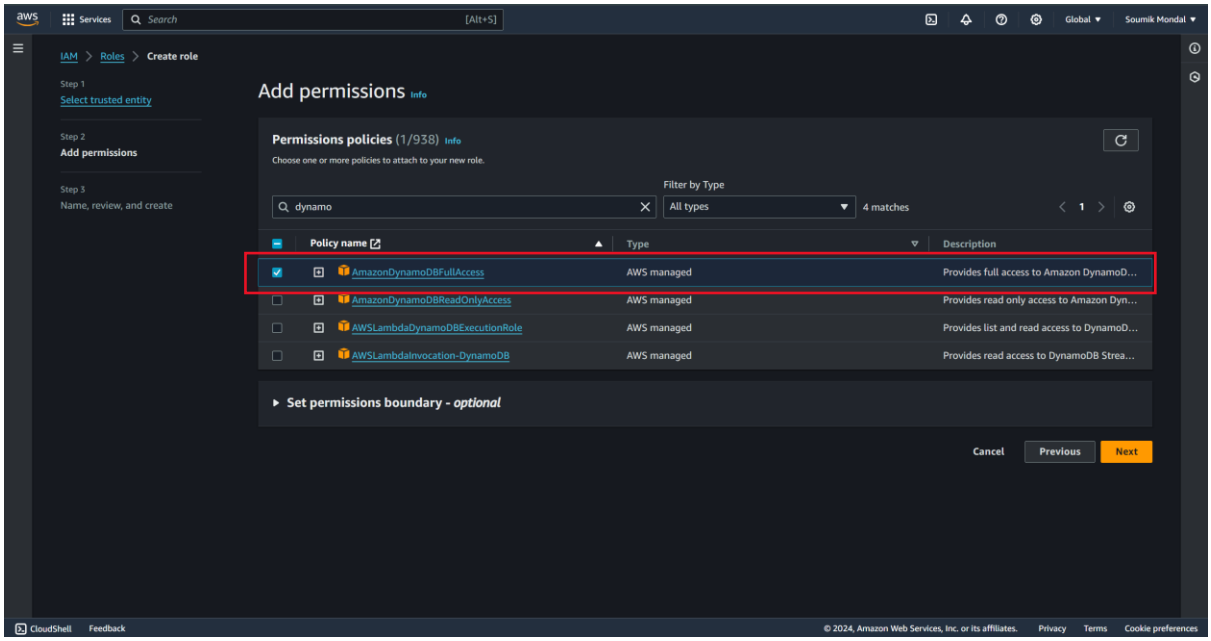


Create partition key as id, and value will be as string

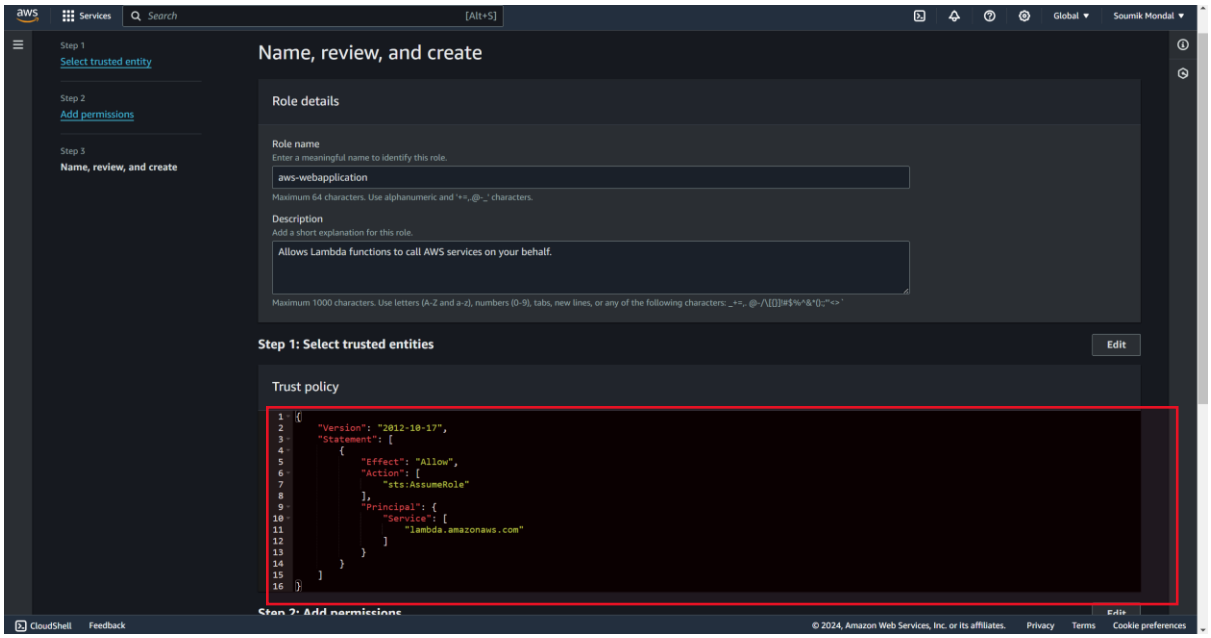
Create an item as Views and give the value as 0

Next, create a role in IAM to give access of DynamoDB to Lambda -



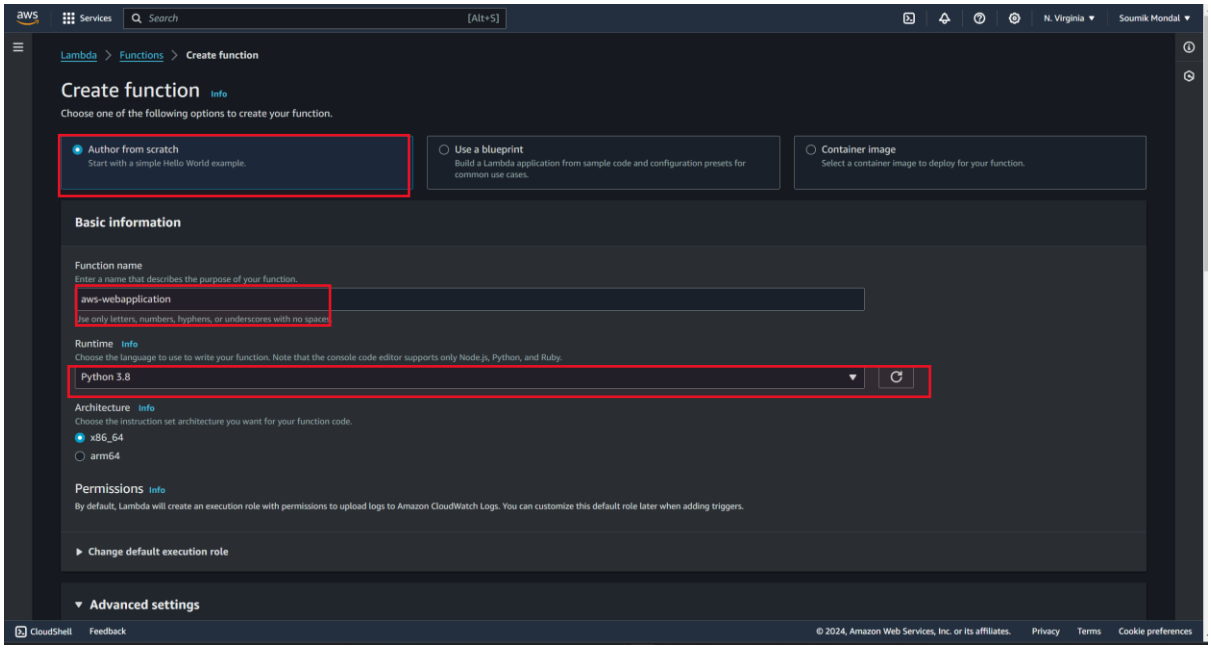


Give full access to DynamoDB

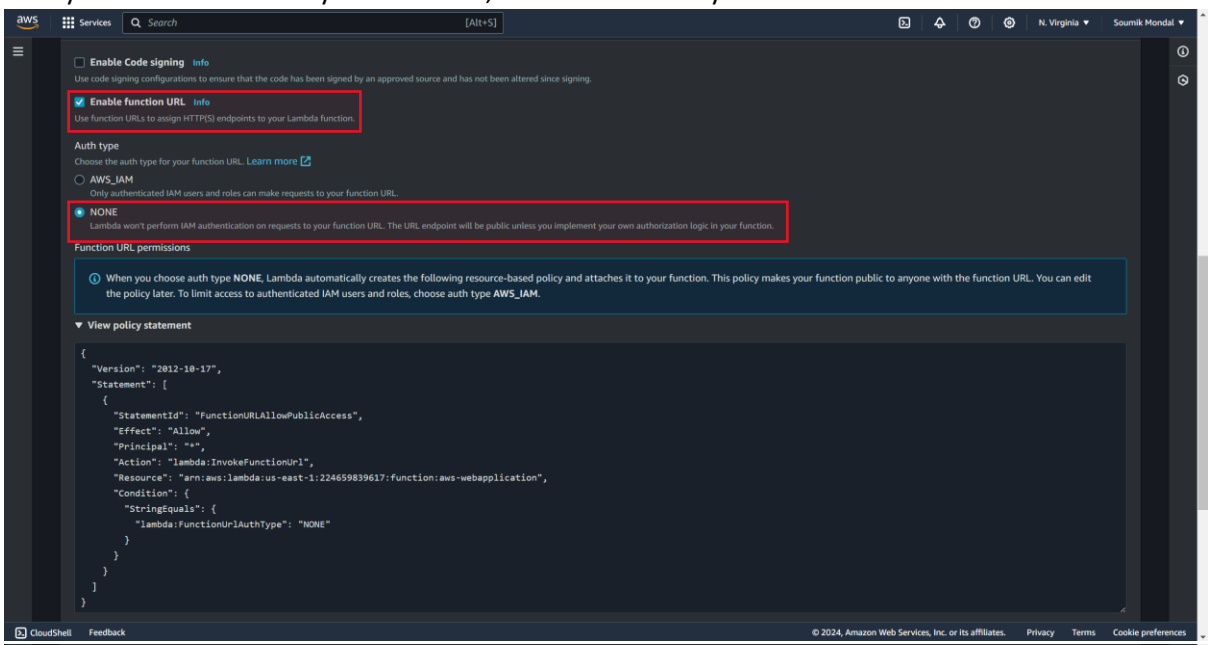


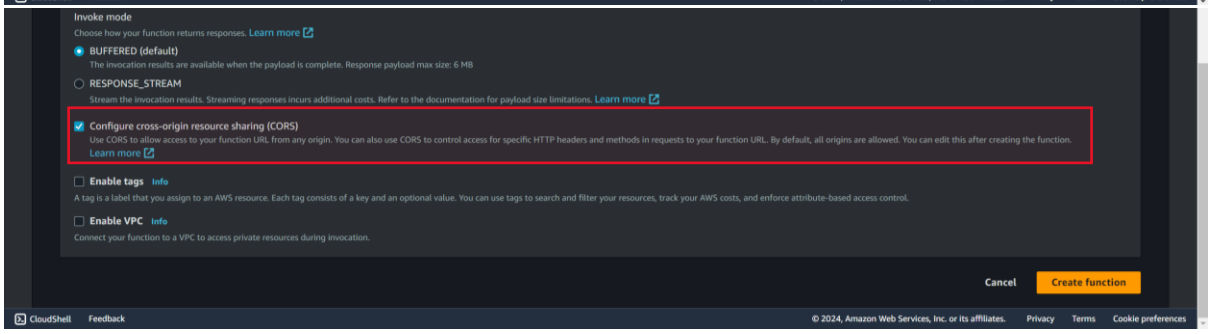
Check that the policy gives full access of DynamoDB to Lambda

Now, create a Lambda function



As My code is written on Python code so, we have to take Python as runtime.

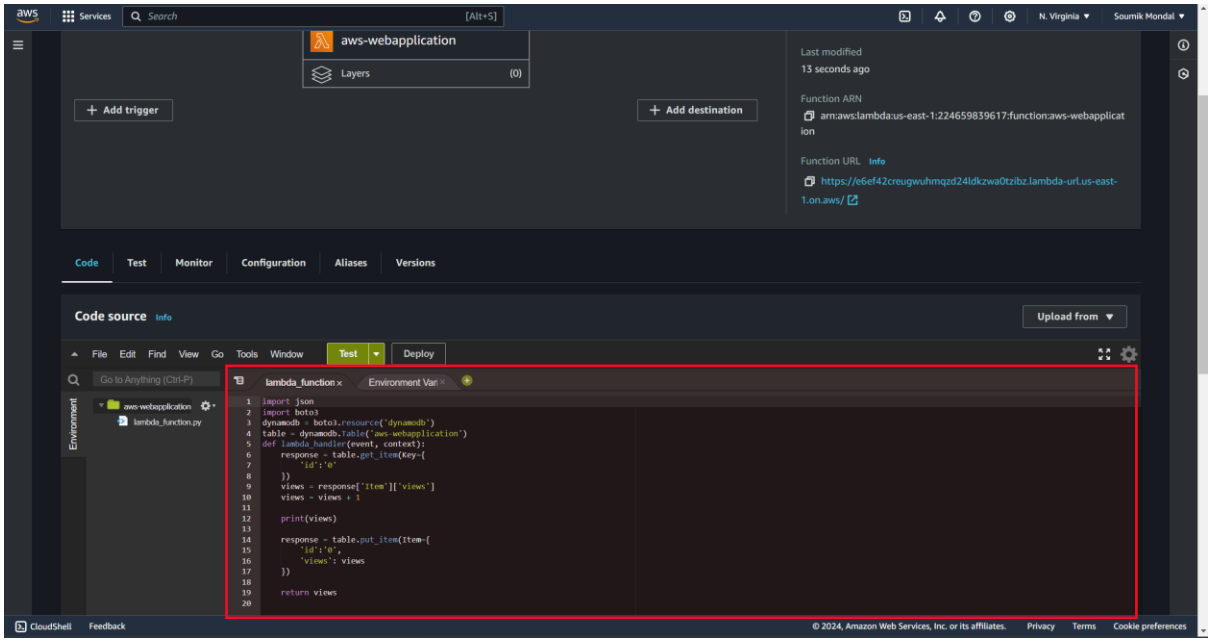


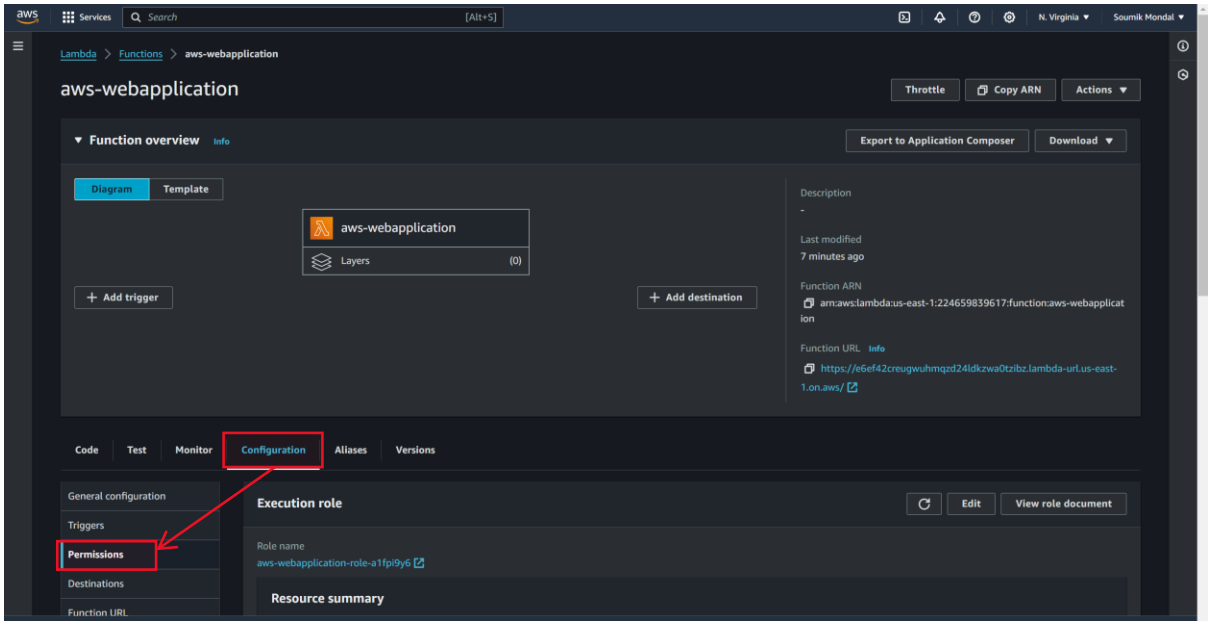


Now paste this code to code editor of lambda -

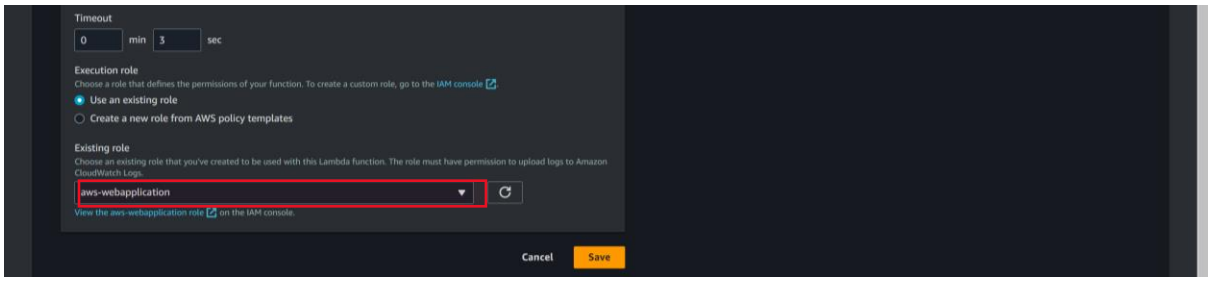
```python
import json
import boto3
dynamodb = boto3.resource('dynamodb')
table = dynamodb.Table('aws-webapplication')
def lambda_handler(event, context):
    response = table.get_item(Key={
        'id':'0'
    })
    views = response['Item']['views']
    views = views + 1

    print(views)

    response = table.put_item(Item={
        'id':'0',
        'views': views
    })

    return views
```

Make sure to replace the table name with your DynamoDB table name



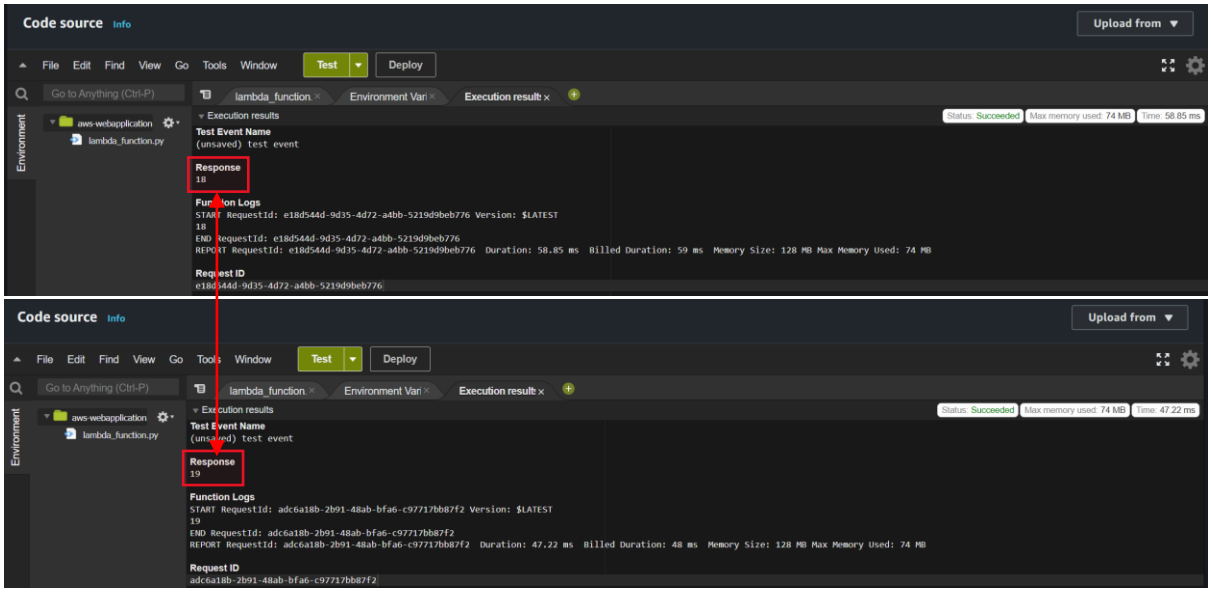


Go to Configuration and then go to the Permission tab



Select the role that you created earlier in IAM and save

Now deploy and test the code -

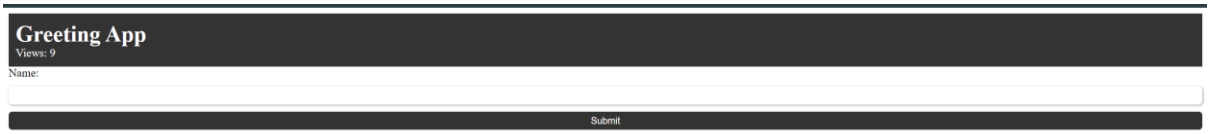


Took multiple tests and successfully got a response

Now edit the script.js file and re-upload to the S3 bucket -

```
const form = document.querySelector('form');
const greeting = document.querySelector('#greeting');

form.addEventListener('submit', (event) => {
    event.preventDefault();
    const name = document.querySelector('#name').value;
    greeting.textContent = `Hello, ${name}!`;
});

const counter = document.querySelector(".counter-number");
async function updateCounter() {
    let response = await fetch(
        "https://e6ef42creugwuhmqzd24ldkzwa0tzibz.lambda-url.us-east-
1.on.aws/"
    );
    let data = await response.json();
    counter.innerHTML = `Views: ${data}`;
}
updateCounter();
```

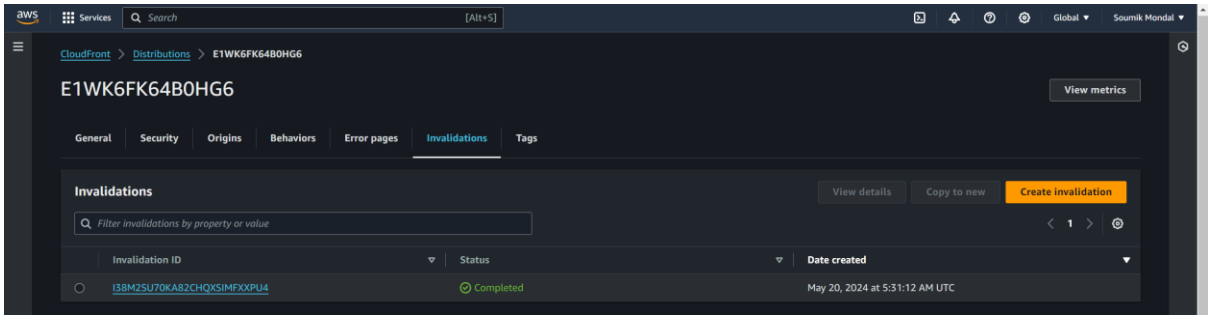
Update the link with function URL, which can be found on Lambda main page



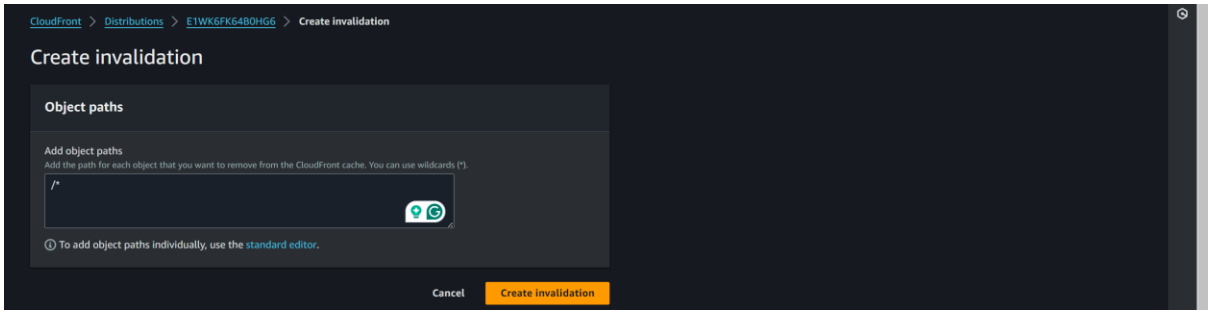
Successfully Fetched the view count, Now every time we refresh the page, the view count will gradually increase

Troubleshoot –

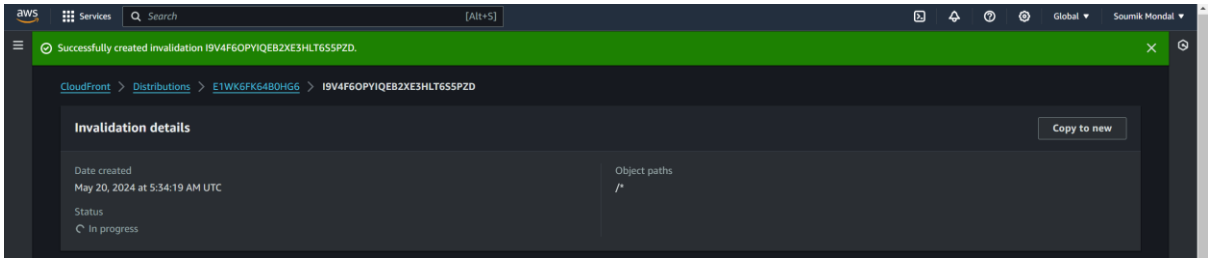
If the view count is not working, we have to create an invalidation and delete the cache of all edge locations.

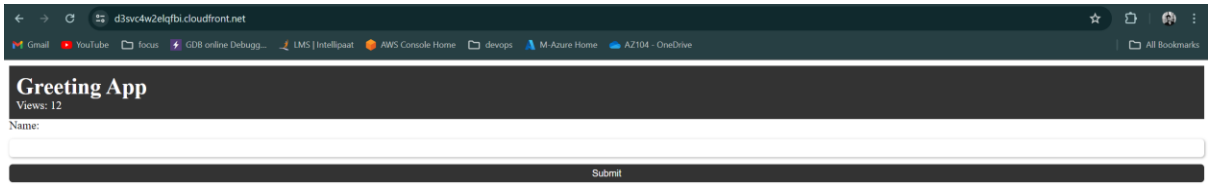


Go to the invalidations tab.



Give "/*" delete all cache in all edge locations





Now, we can access the webpage and view count seamlessly.

N.B. - After this demo, make sure to delete all resources

----------completed----------