**Project Capstone**

To enhance the performance of the restaurant application, a cloud-based architecture is designed using AWS services. Static content, including images, HTML, CSS, and JavaScript, will be stored in an S3 bucket with static website hosting enabled. AWS CloudFront will serve as a Content Delivery Network (CDN) to cache and deliver static content globally, reducing latency. AWS Route 53 will handle DNS-based load balancing, efficiently distributing traffic. The backend application will be deployed on AWS EC2 instances for scalable processing. Additionally, governance and cost management will be ensured through AWS Organizations and IAM policies for access control, AWS Budgets and Cost Explorer for cost monitoring, and a tagging strategy for environment-based billing. Furthermore, Google TNT Drive will be used to map the S3 bucket to the EC2 instances, optimizing storage access and performance.

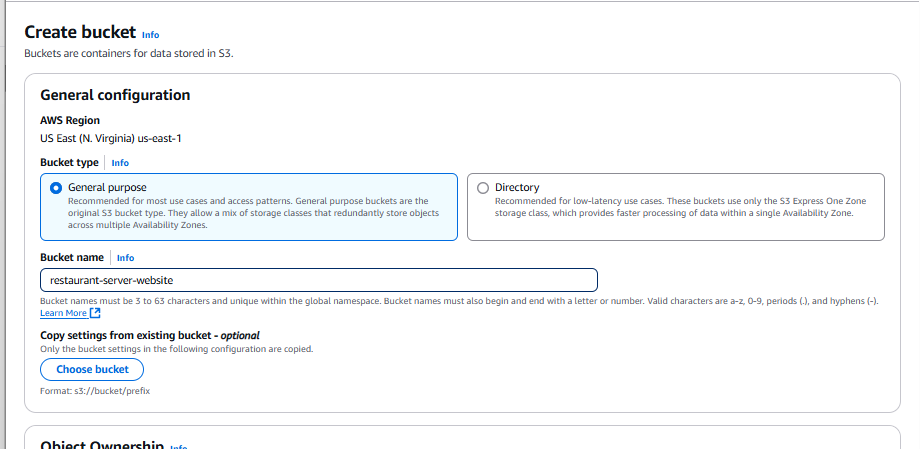
**Step 1: Host Static Content on Amazon S3**

**1.1 Create an S3 Bucket**

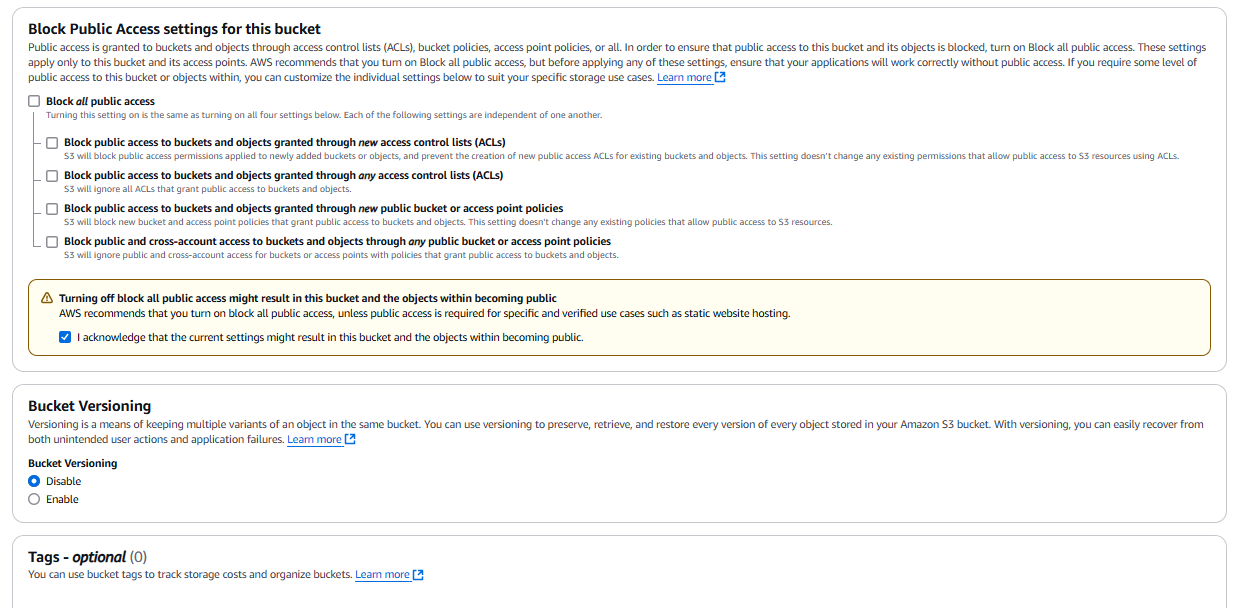
1. Log in to the **AWS Management Console**.
2. Navigate to **Amazon S3** and click **Create bucket**.



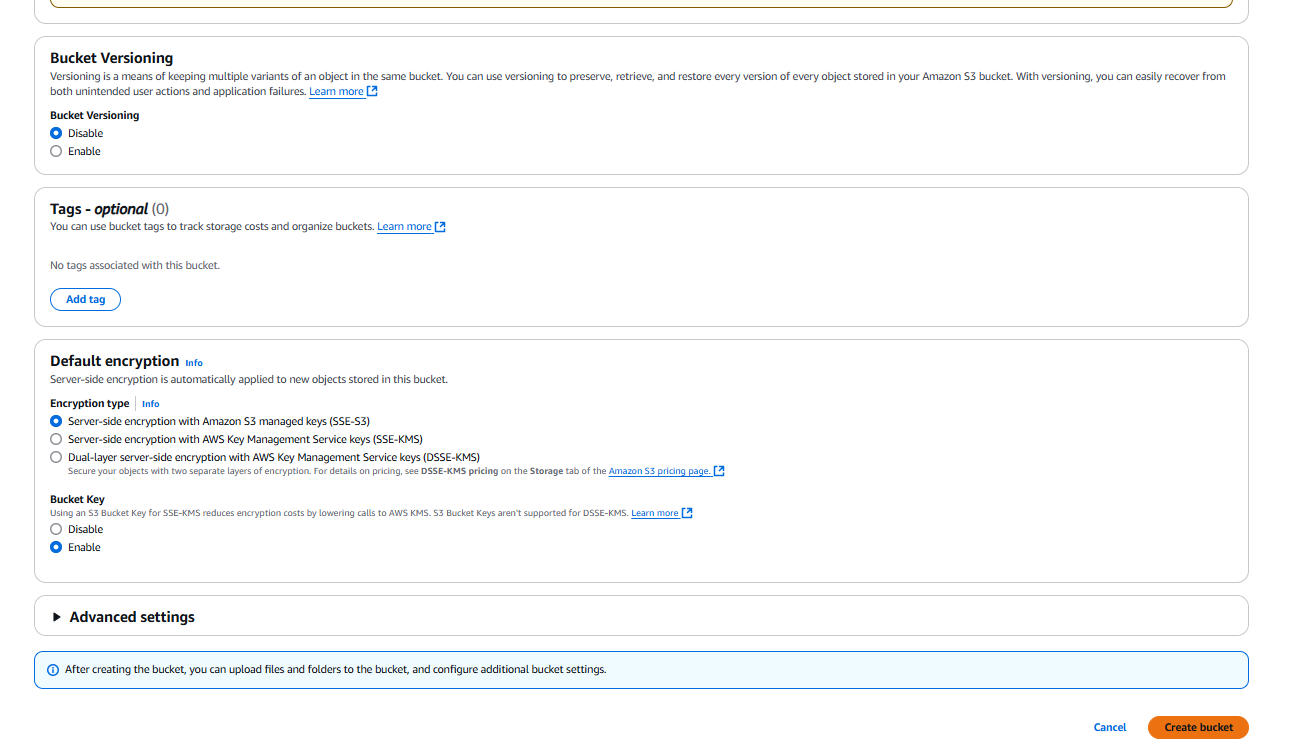
1. Enter a **unique bucket name** (e.g., restaurant-website-serve).

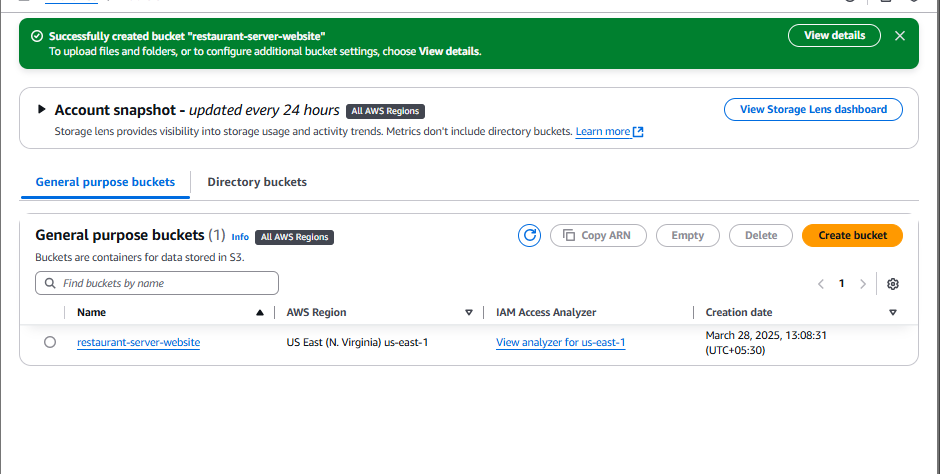


1. Uncheck **Block all public access** (since this is a public-facing website).



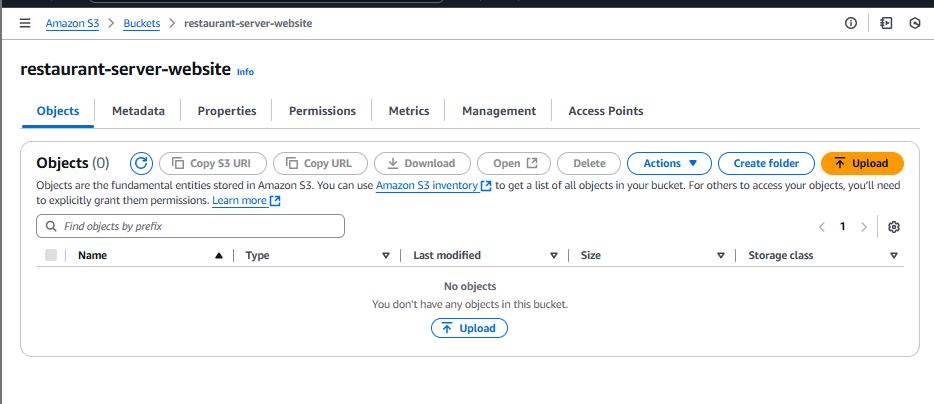
1. Click **Create bucket**.



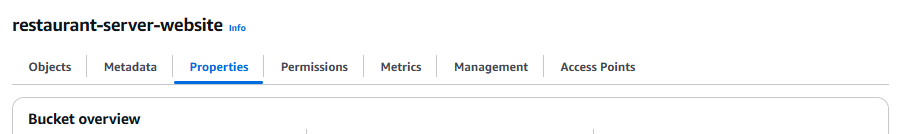
  
Storage Bucket successfully created.

**1.2 Enable Static Website Hosting**

1. Open the **S3 bucket** you just created.



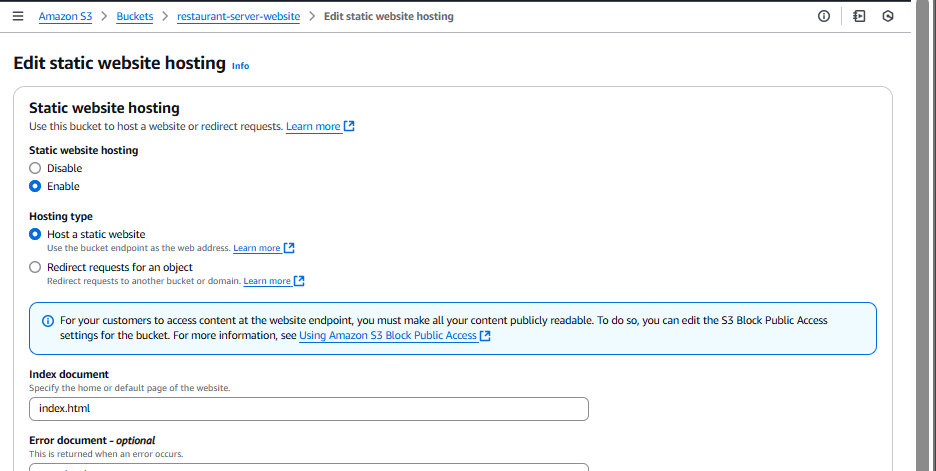
1. Go to the **Properties** tab and locate **Static website hosting**.



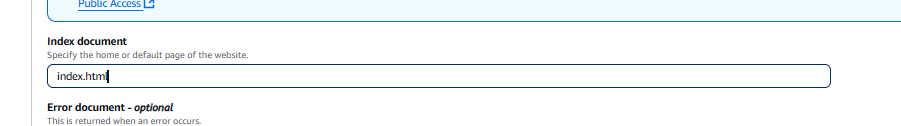
1. Select **Enable** and choose **Host a static website**.



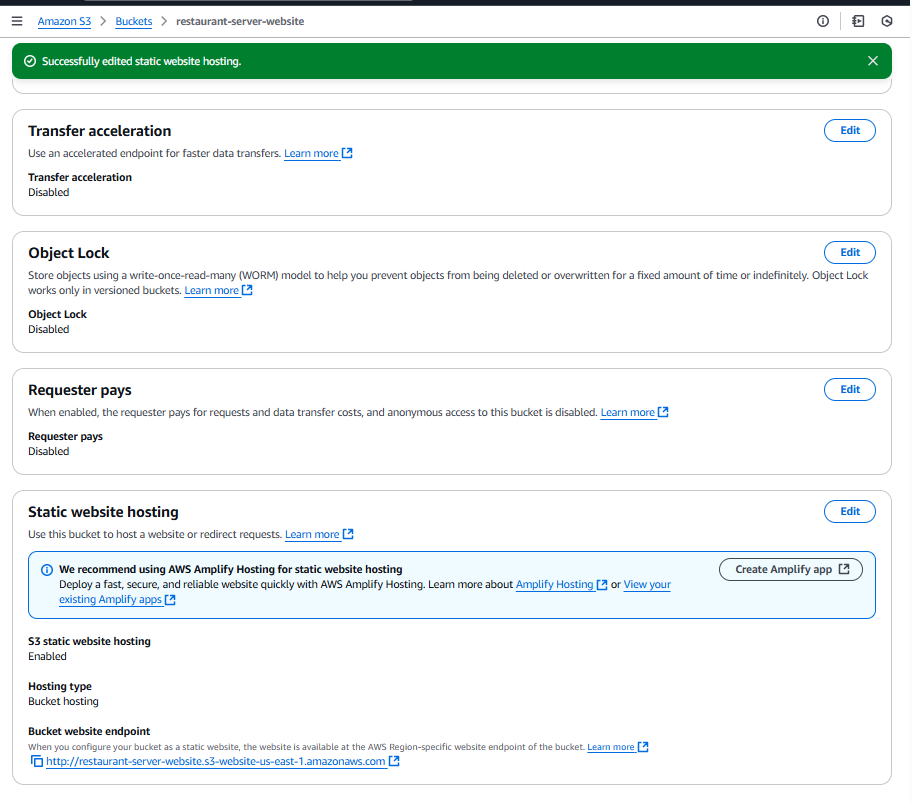
After making it enable



1. Set the **index document** (e.g., index.html) and click create bucket

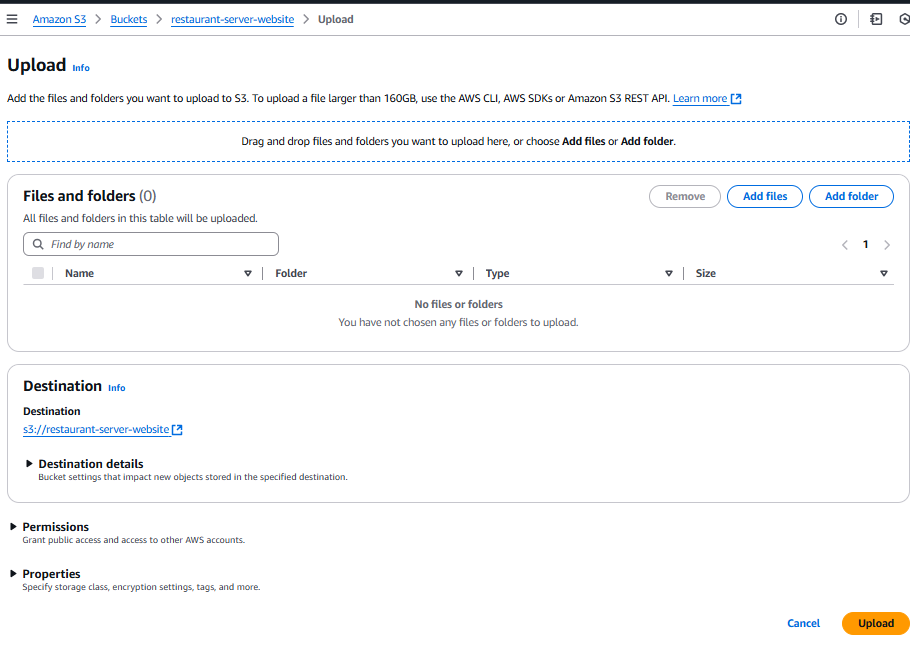


1. Note the **bucket endpoint URL** (this will be used later).

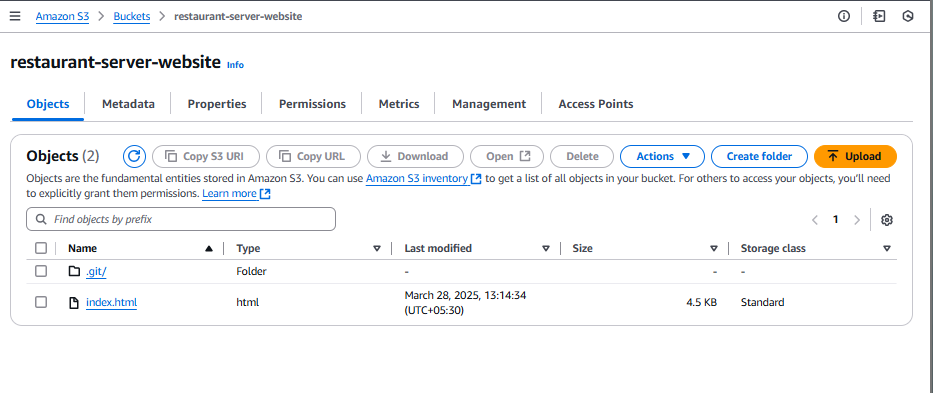


**1.3 Upload Static Assets**

1. Open your **S3 bucket**.



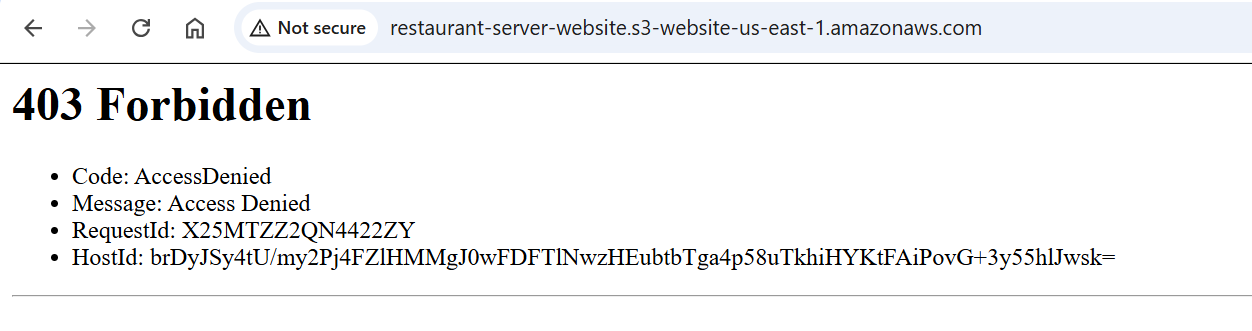
1. Click **Upload** and select your website files by dragging and dropping them here and click upload.



Files uploaded successfully as shown in the above screenshot.

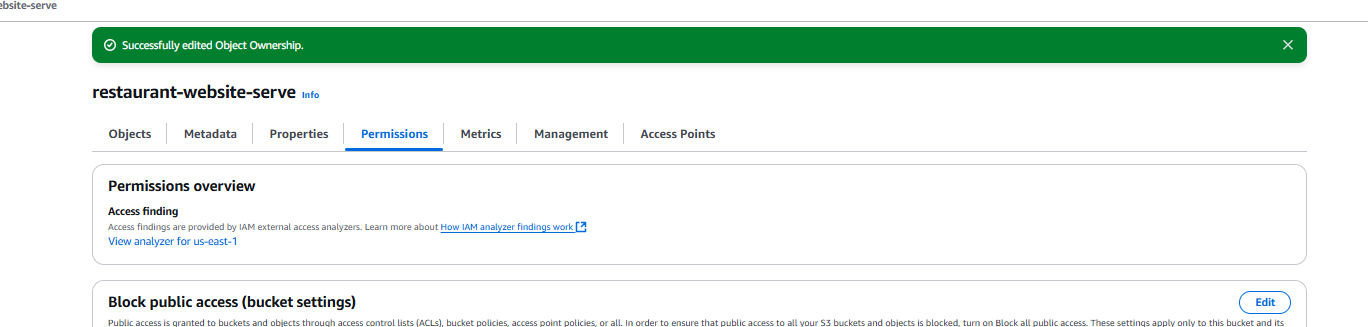
1. Ensure **public read access** is enabled for these files.

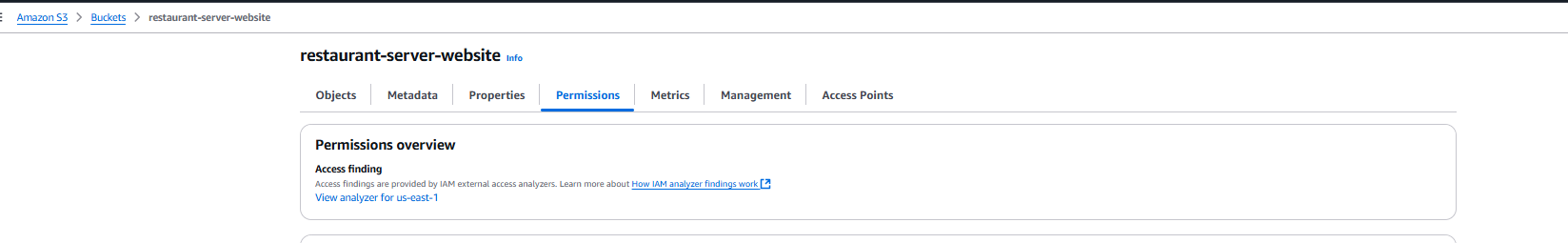
Access the bucket url: <http://restaurant-server-website.s3-website-us-east-1.amazonaws.com>

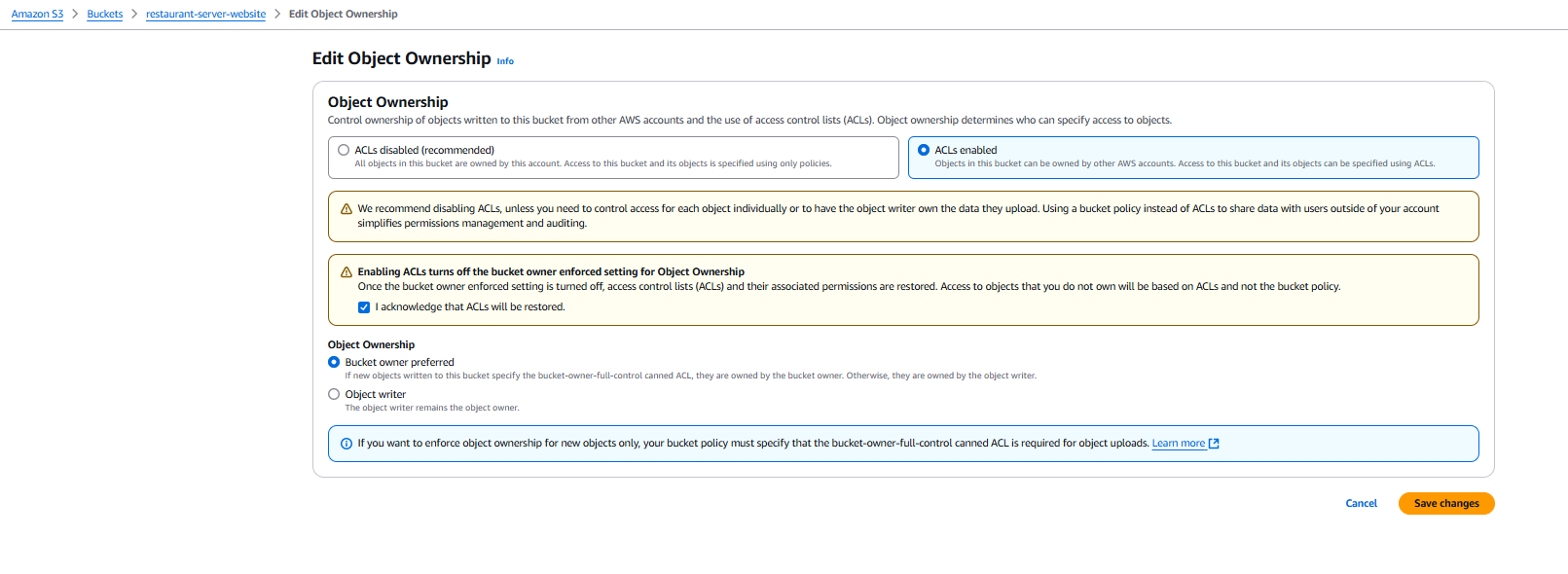


The website is not yet open to public access. For this, we need to edit the Object Ownership of the bucket

Now save changes

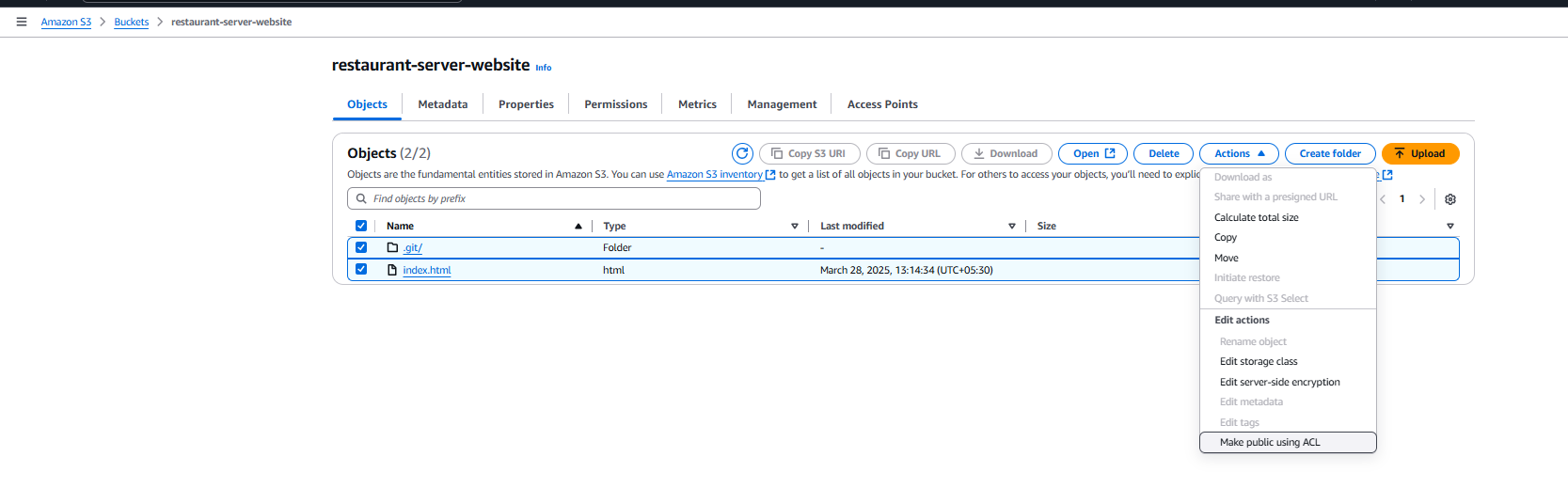


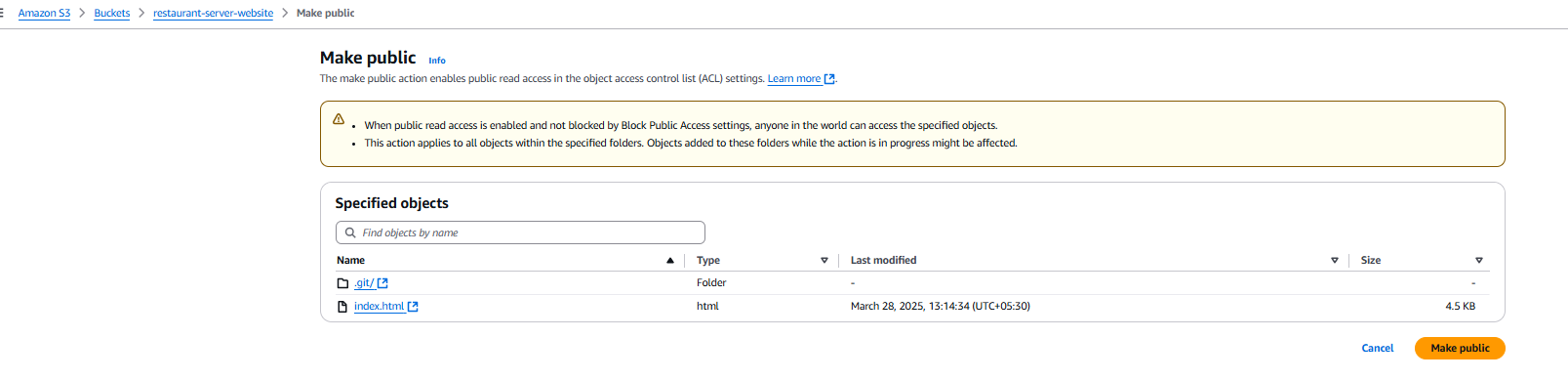
1. Go to Permissions and edit the object ownership   
   

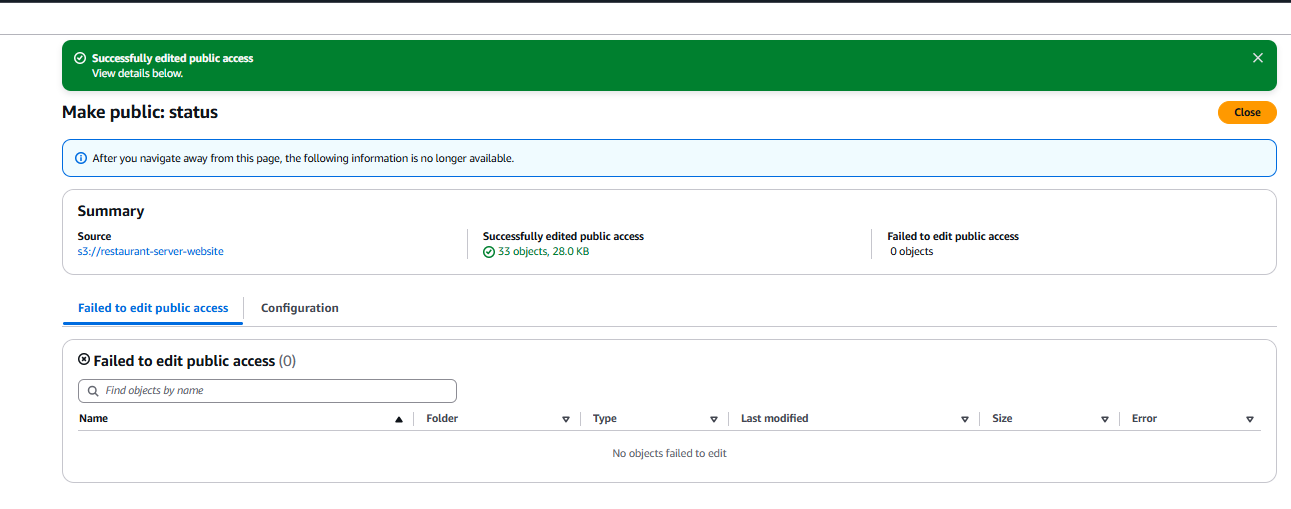




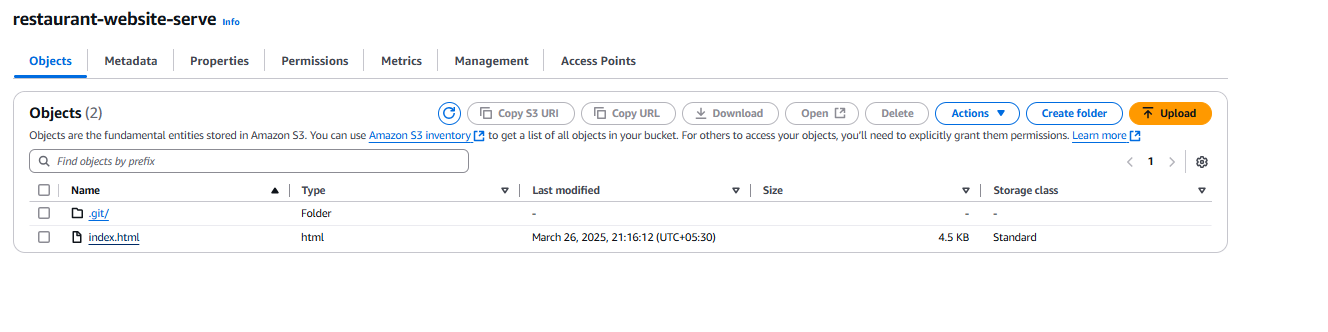
1. Now again go to the bucket we created and open the bucket then To make the objects publicly accessible go to actions and make Public using ACL



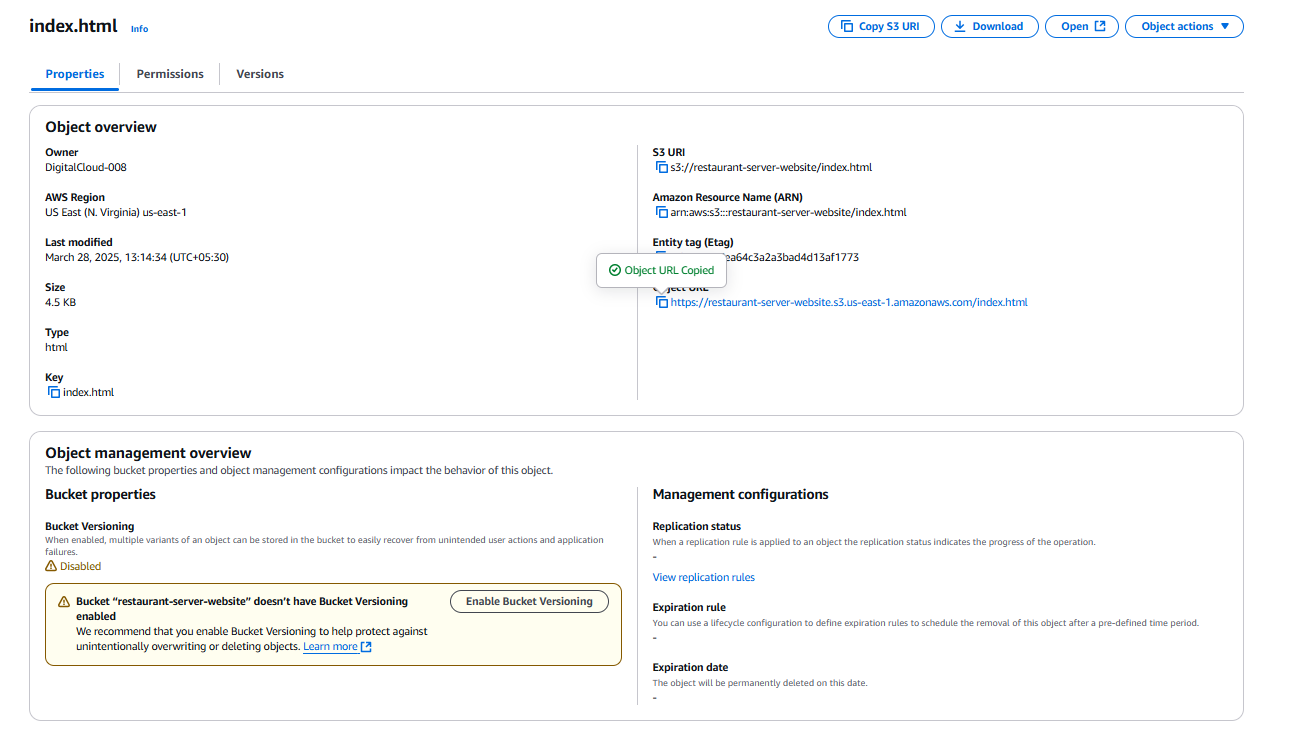


Objects for the website successfully uploaded in the s3 which are accessible over internet

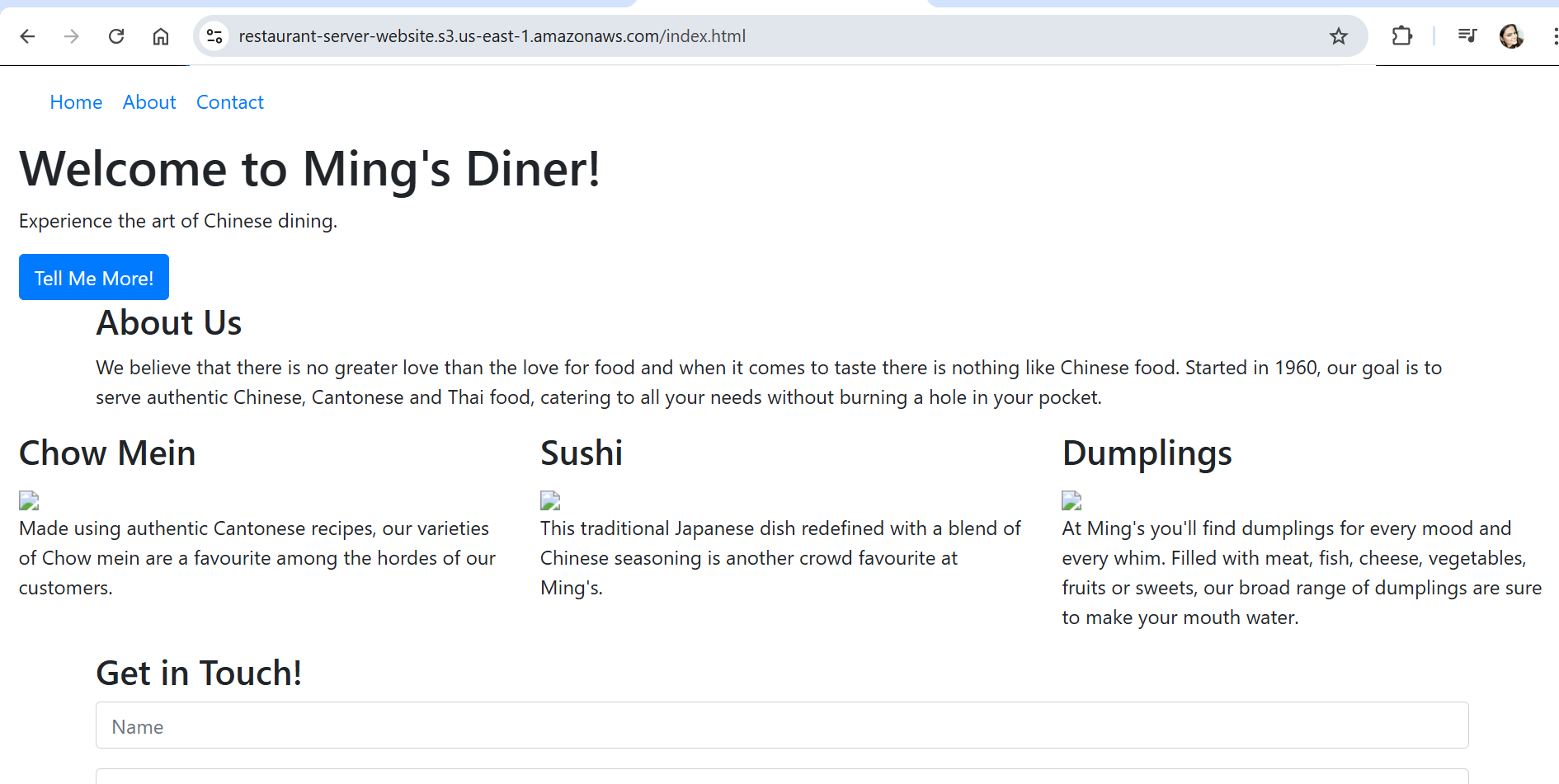
Now go to Index.html and access



and access the website at : <https://restaurant-server-website.s3.us-east-1.amazonaws.com/index.html> as shown below:



Hence the website is successfully hosted over the internet.

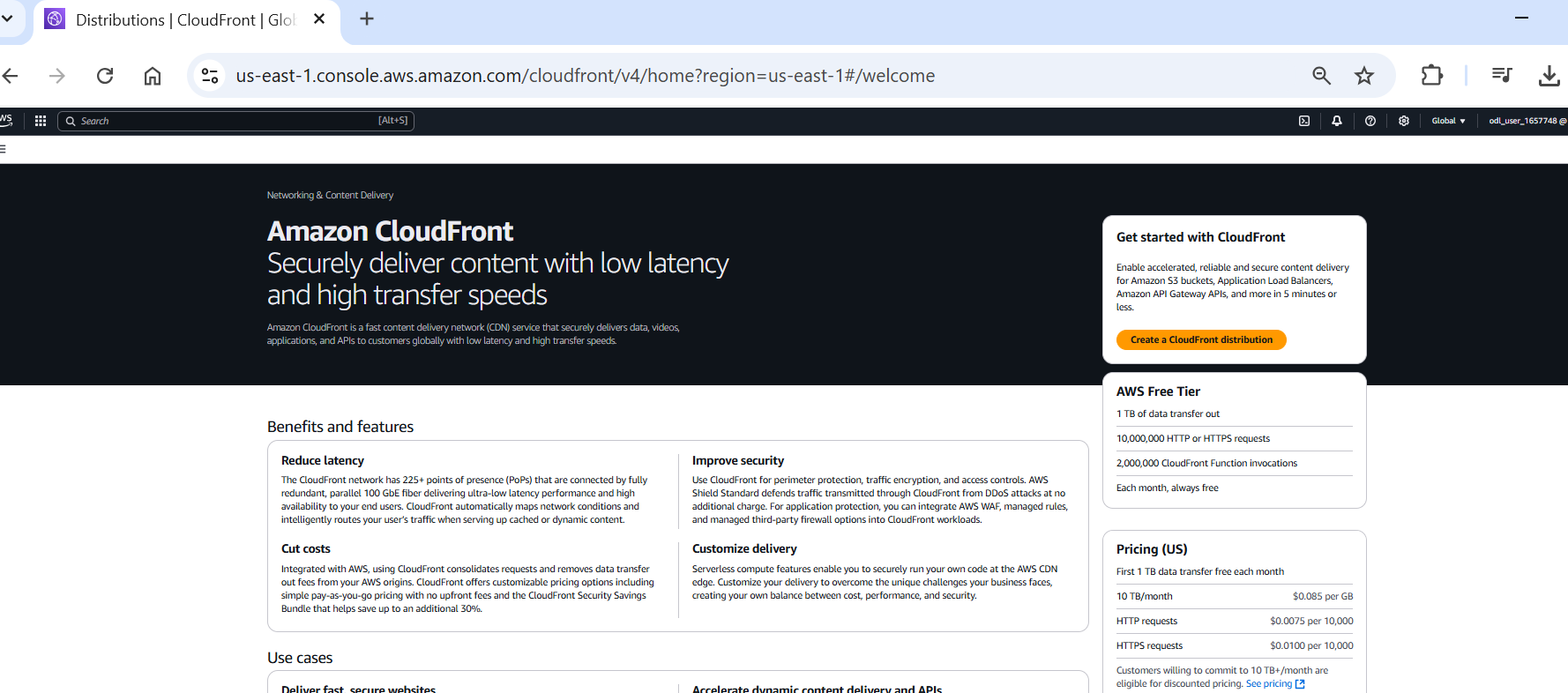


Now the static Website is successfully created on S3.

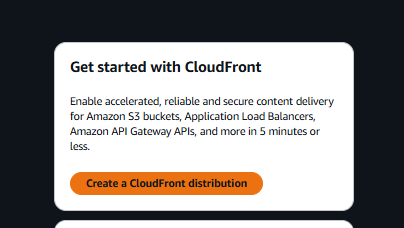
**Step 2: Create a CloudFront Distribution**

**2.1 Set Amazon S3 as the Origin**

1. Navigate to **AWS CloudFront**.



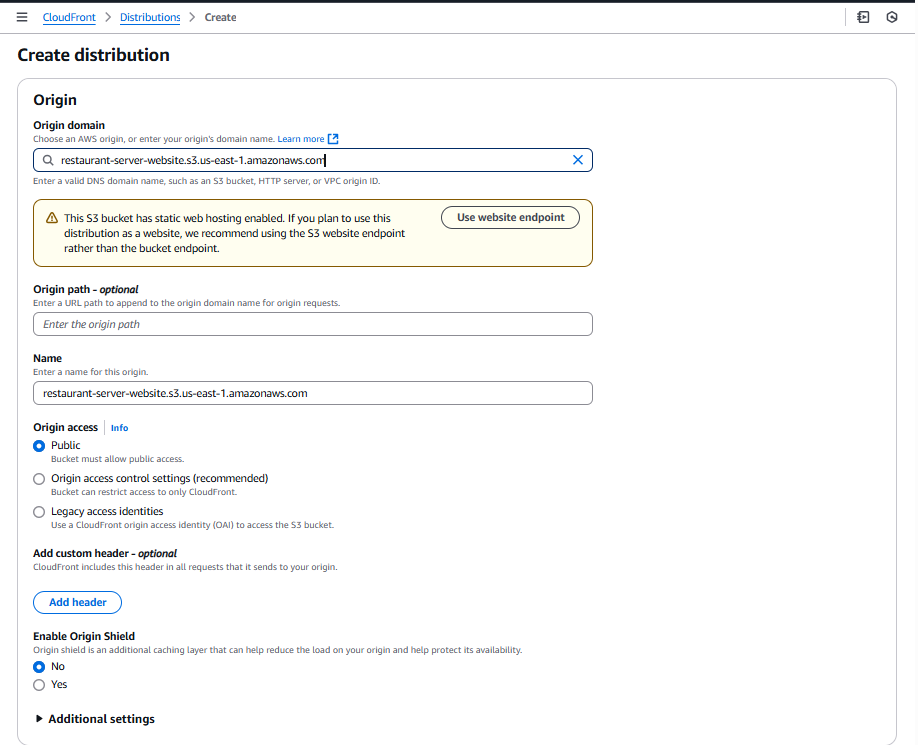
1. Click **Create Distribution**.

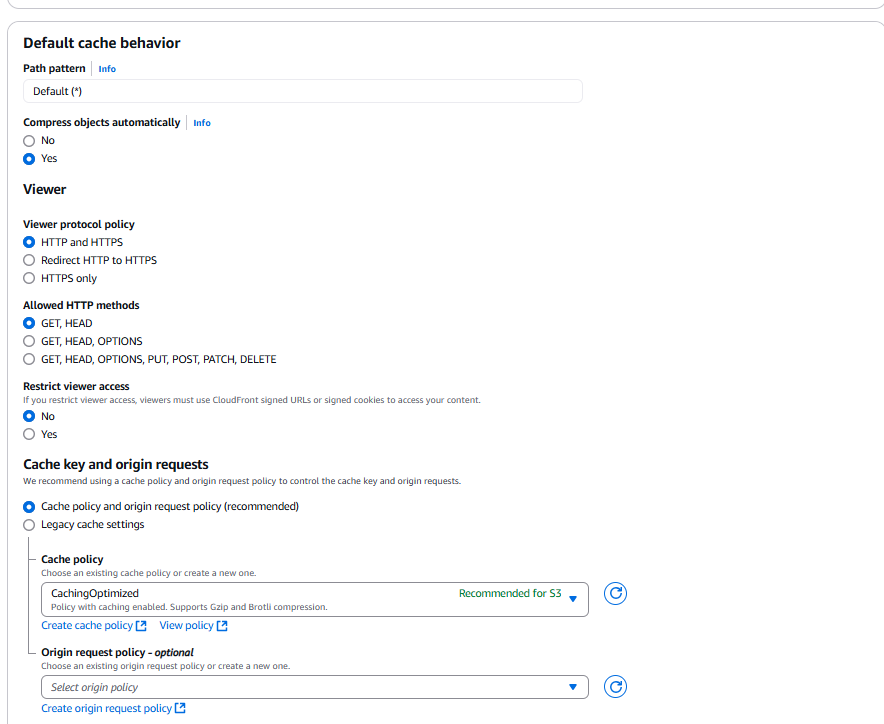


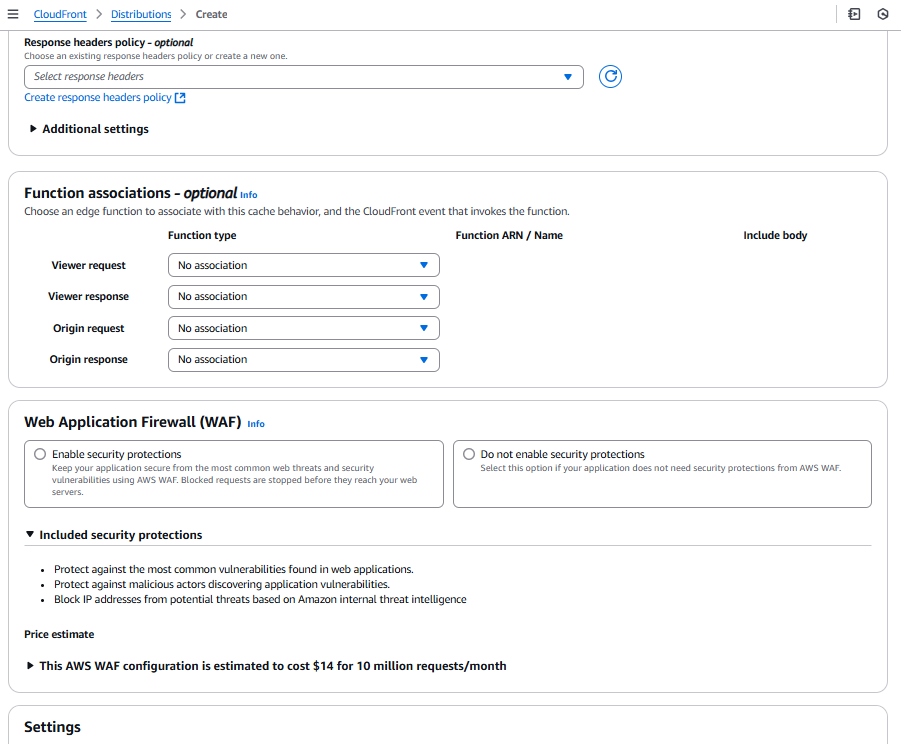
1. Under **Origin Settings**, select **Amazon S3 bucket** and choose the S3 bucket created earlier.

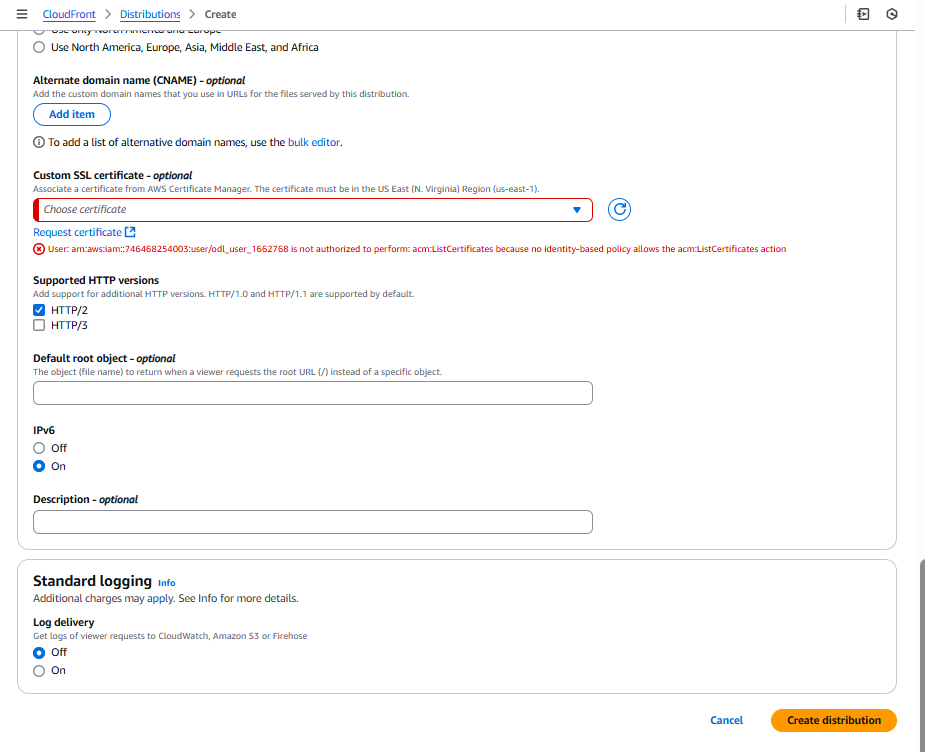


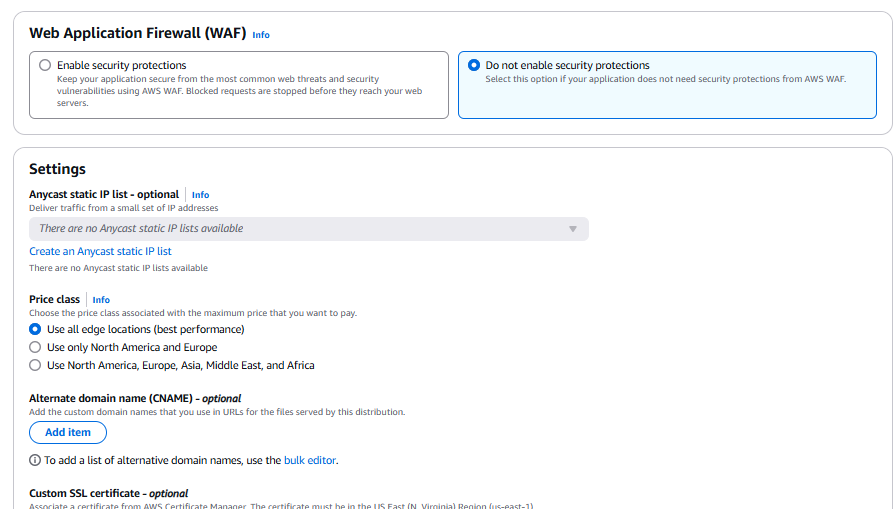
Keep Everything Default as shown below:





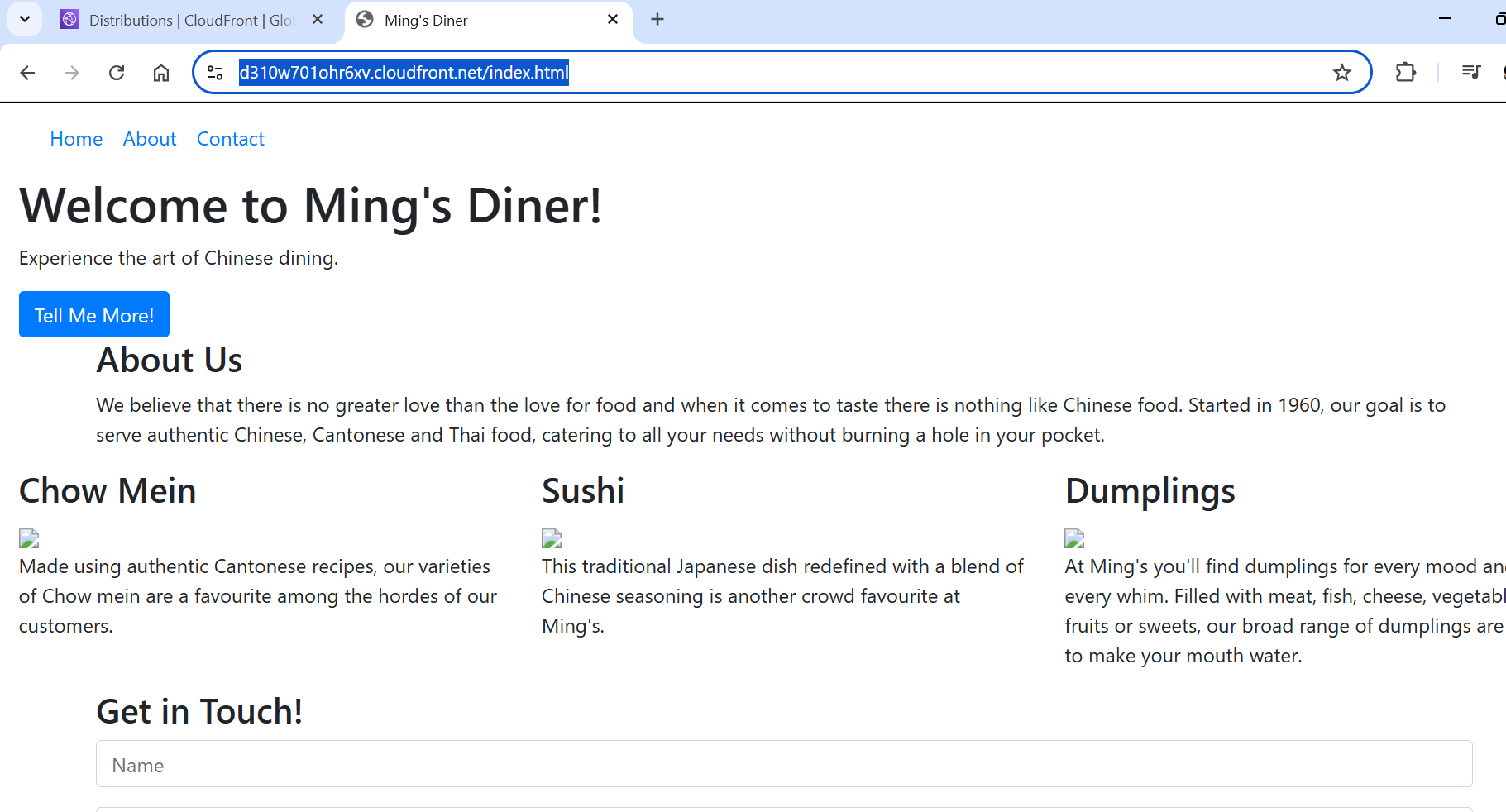




Do not Enable WAF for security   


Click **Create Distribution to**  Save and deploy the CloudFront distribution.

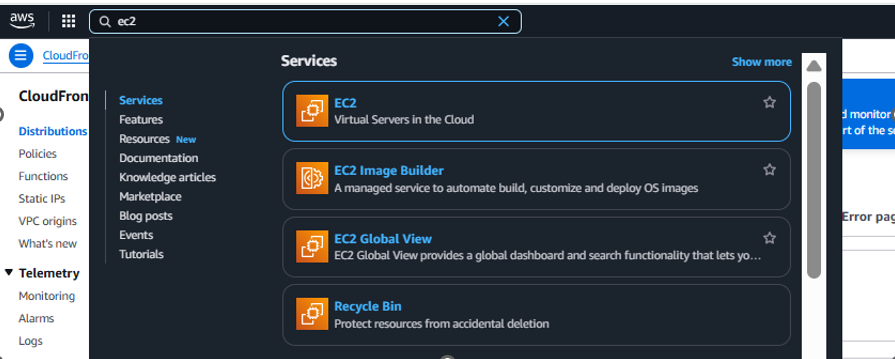
Hence, the website is accessible over CloudFront at the domain <https://d310w701ohr6xv.cloudfront.net/index.html>



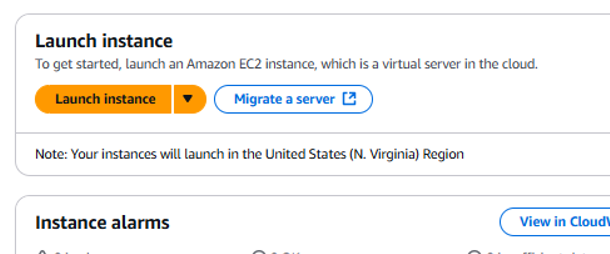
**Step 3: Deploy the Backend Application on EC2**

**3.1 Launch an EC2 Instance**

1. Navigate to **EC2 Dashboard** in the AWS Console.

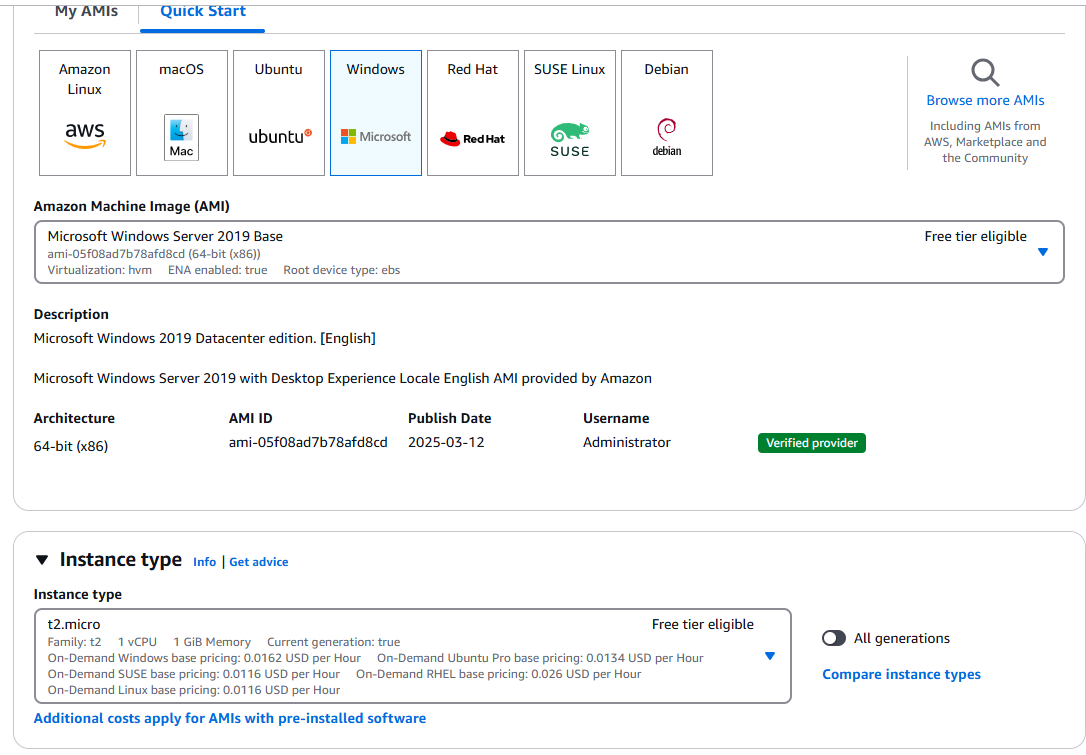


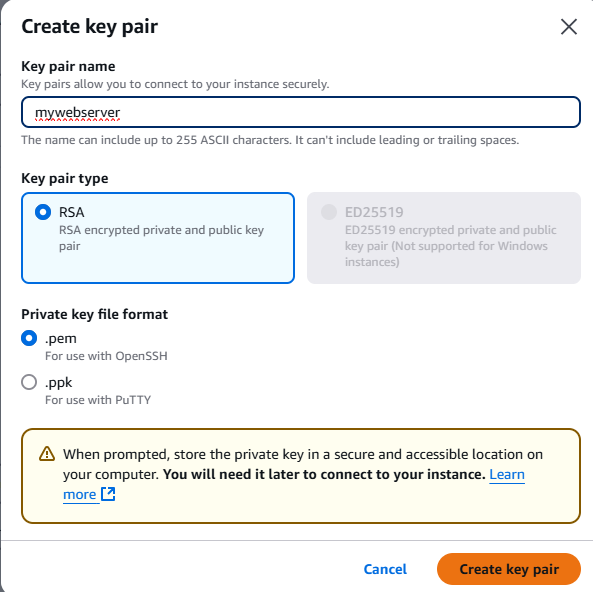
1. Click **Launch Instance**.

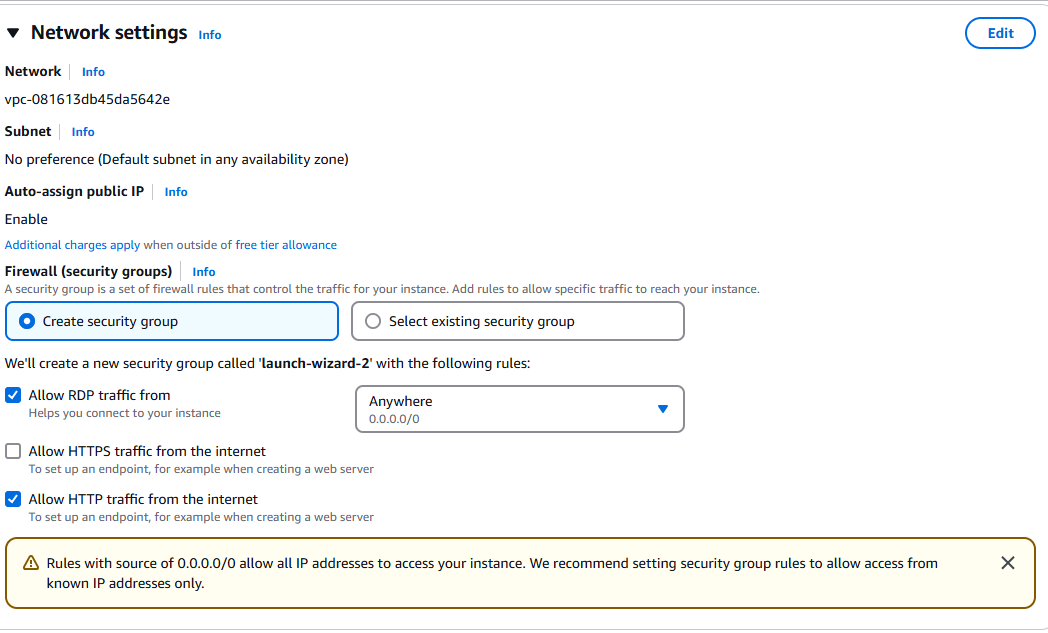


**Name the instance as below:  
**

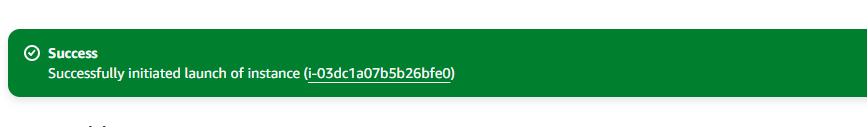
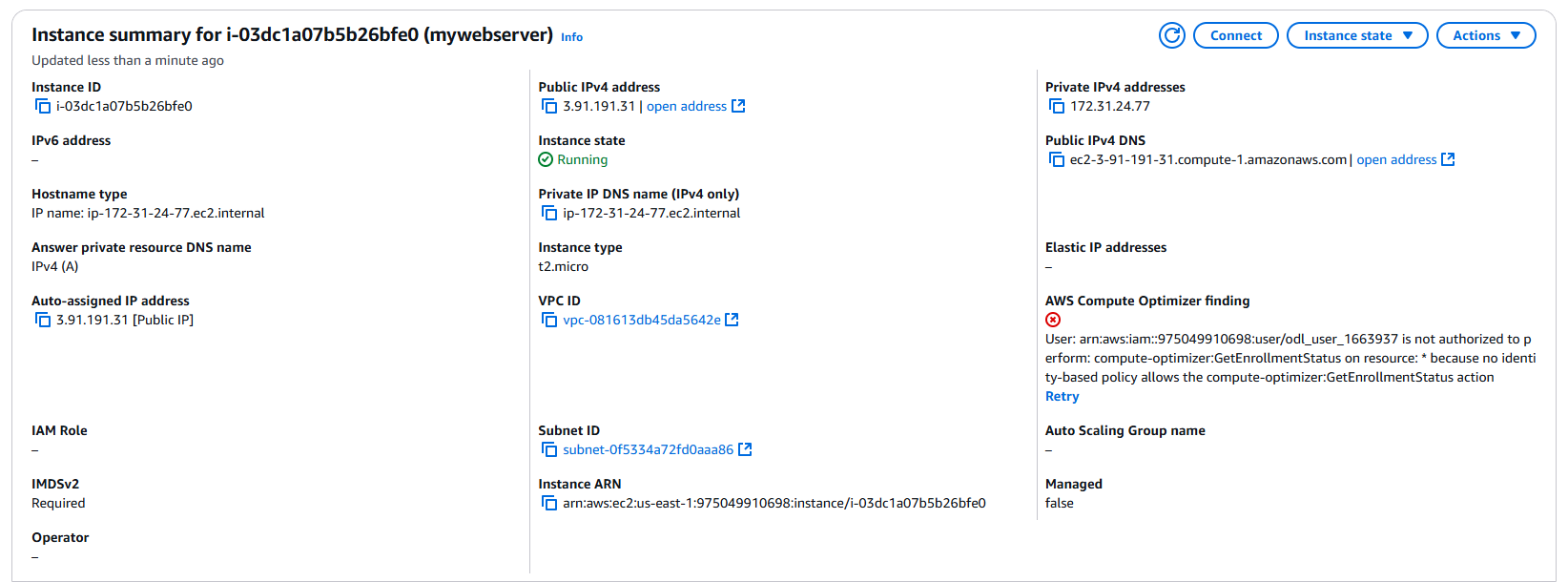
1. Select windows AMI and t3.micro as the **instance type**



Create a key pair:  


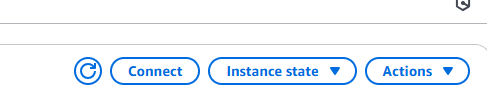
**Create security group:**  


Click **Launch** to create instance.

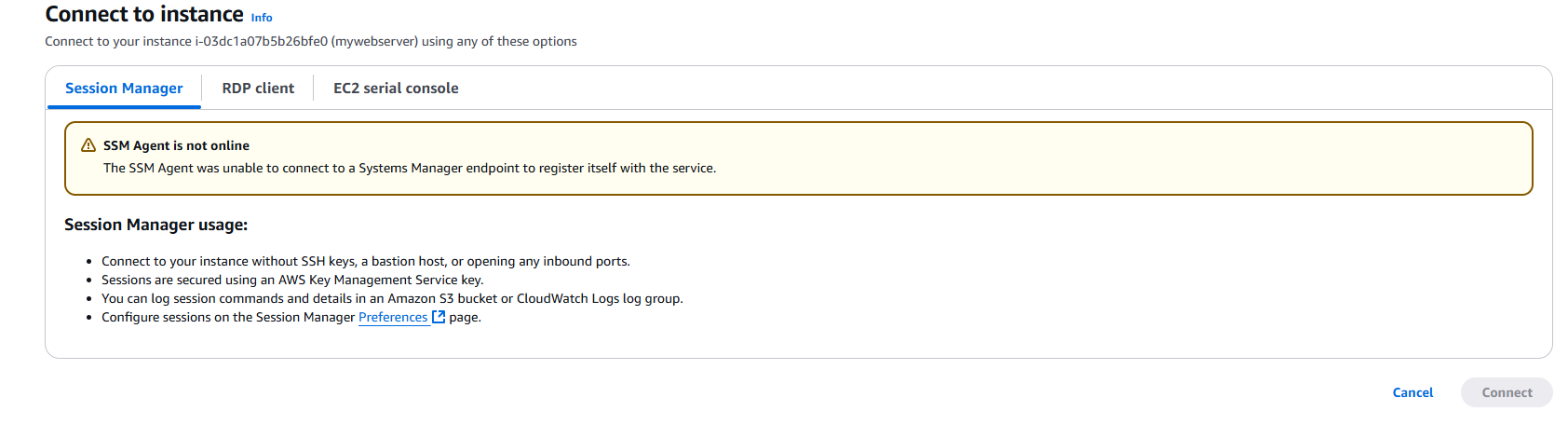
Instance created.  


**3.2 Install a Web Server and Deploy the Application**

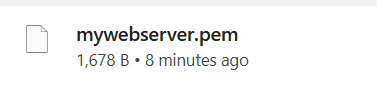
**1. To connect to the EC2 CLICK on the connect as shown**

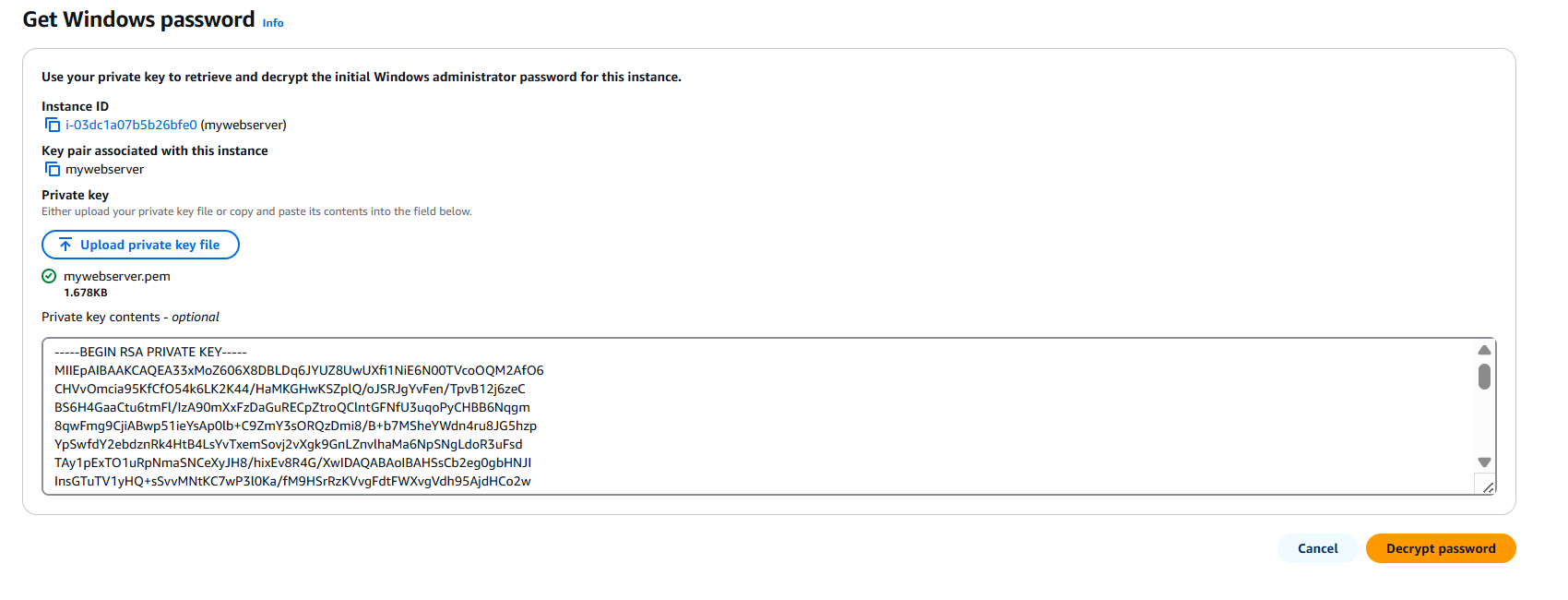
****

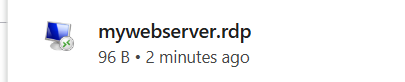
**3. the following screen will appear:**

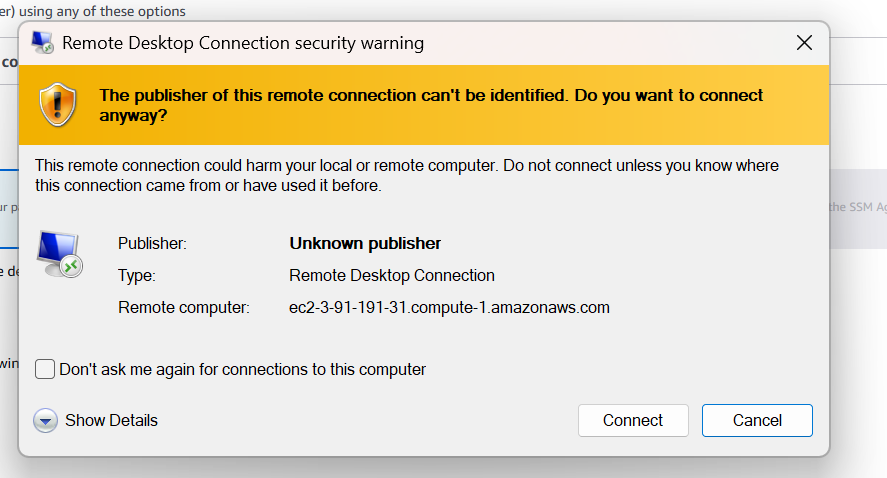
****

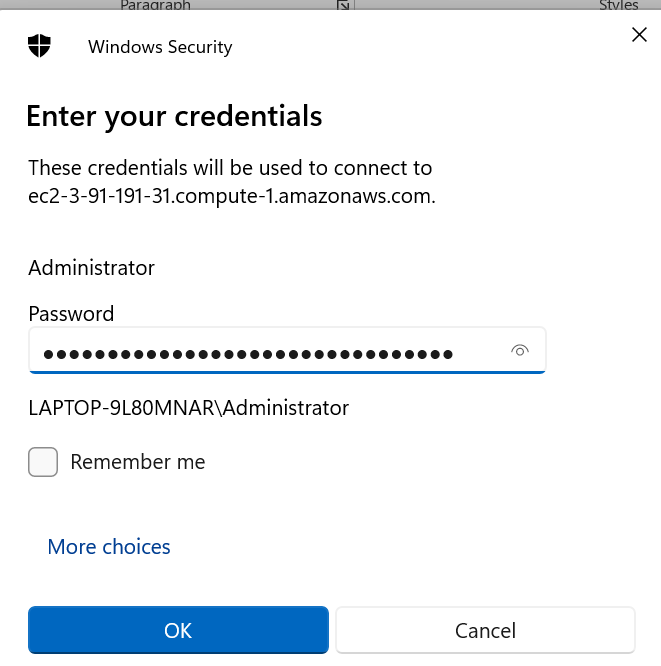
**Click on RDP CLIENT  
**

**To get windows passwords get the password by uploading private key pair created previously  
**

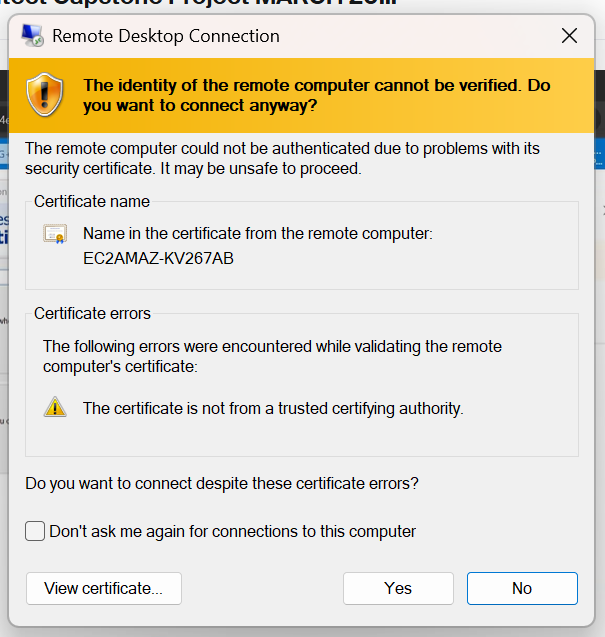
**Now decrypt the key  
**

**Now go to the virtual machine and connect  
**

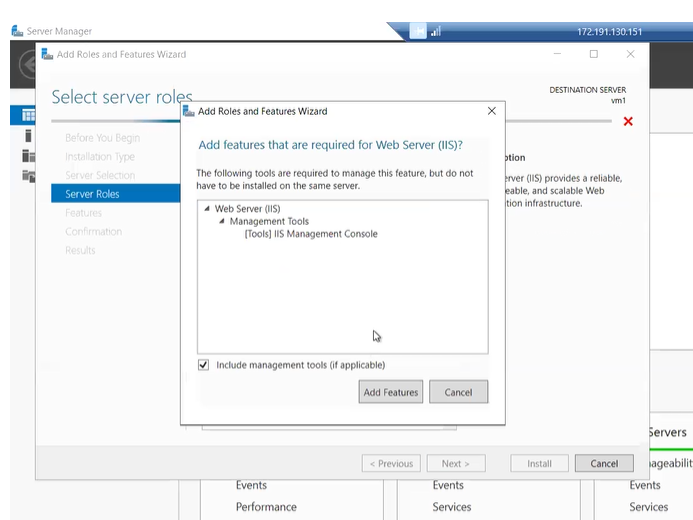
****

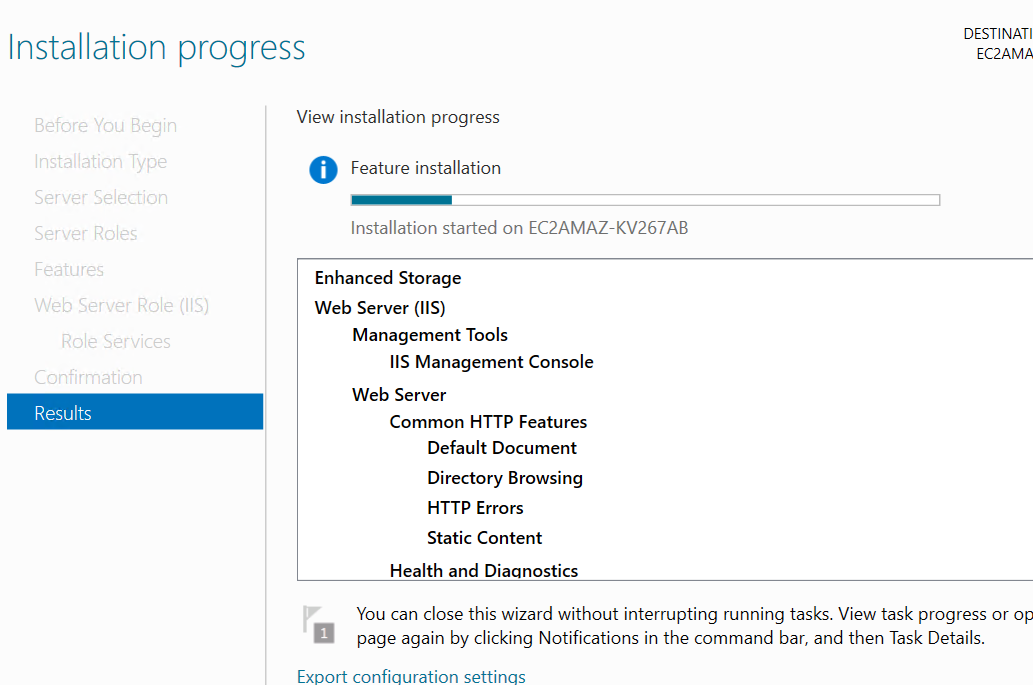
****

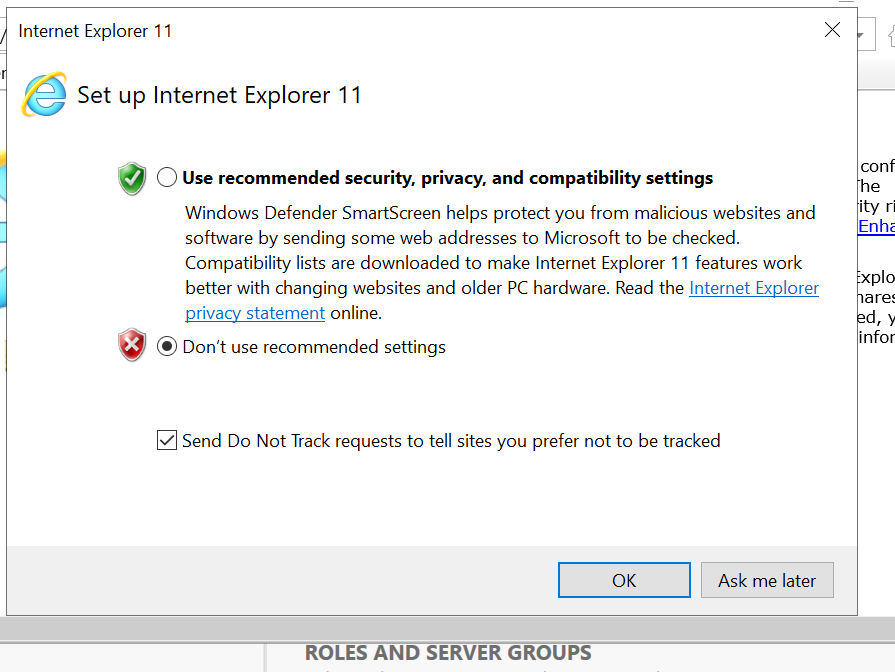
**Click yes**

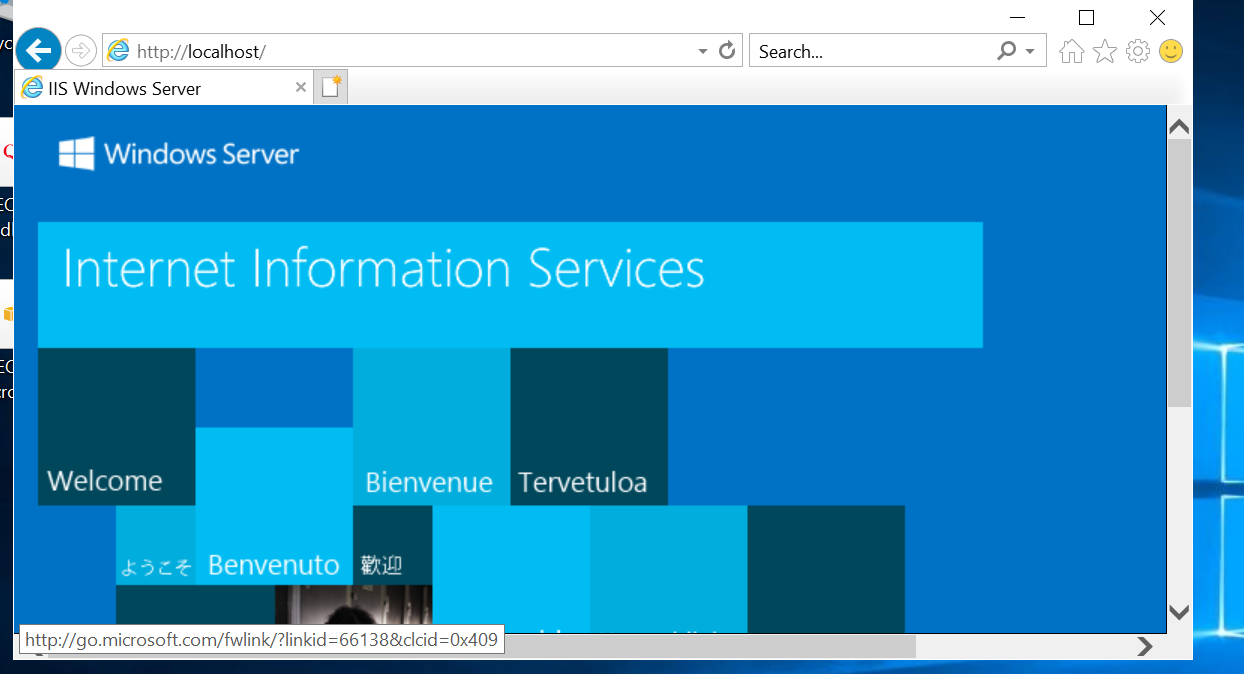
****

**Vm created and deployed  
**

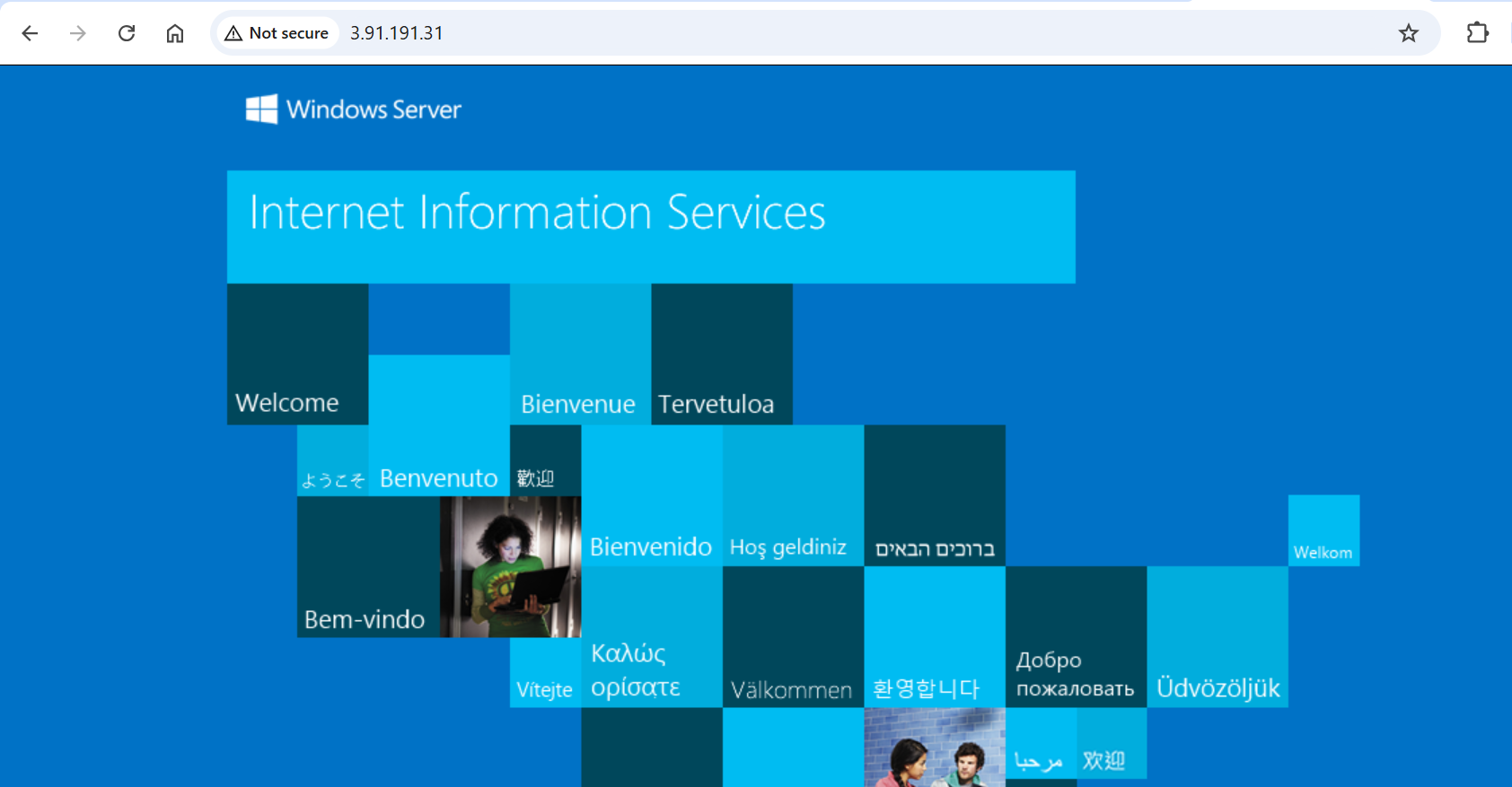
**Now install the local server roles and features   
**

**And finally install IIS  
**

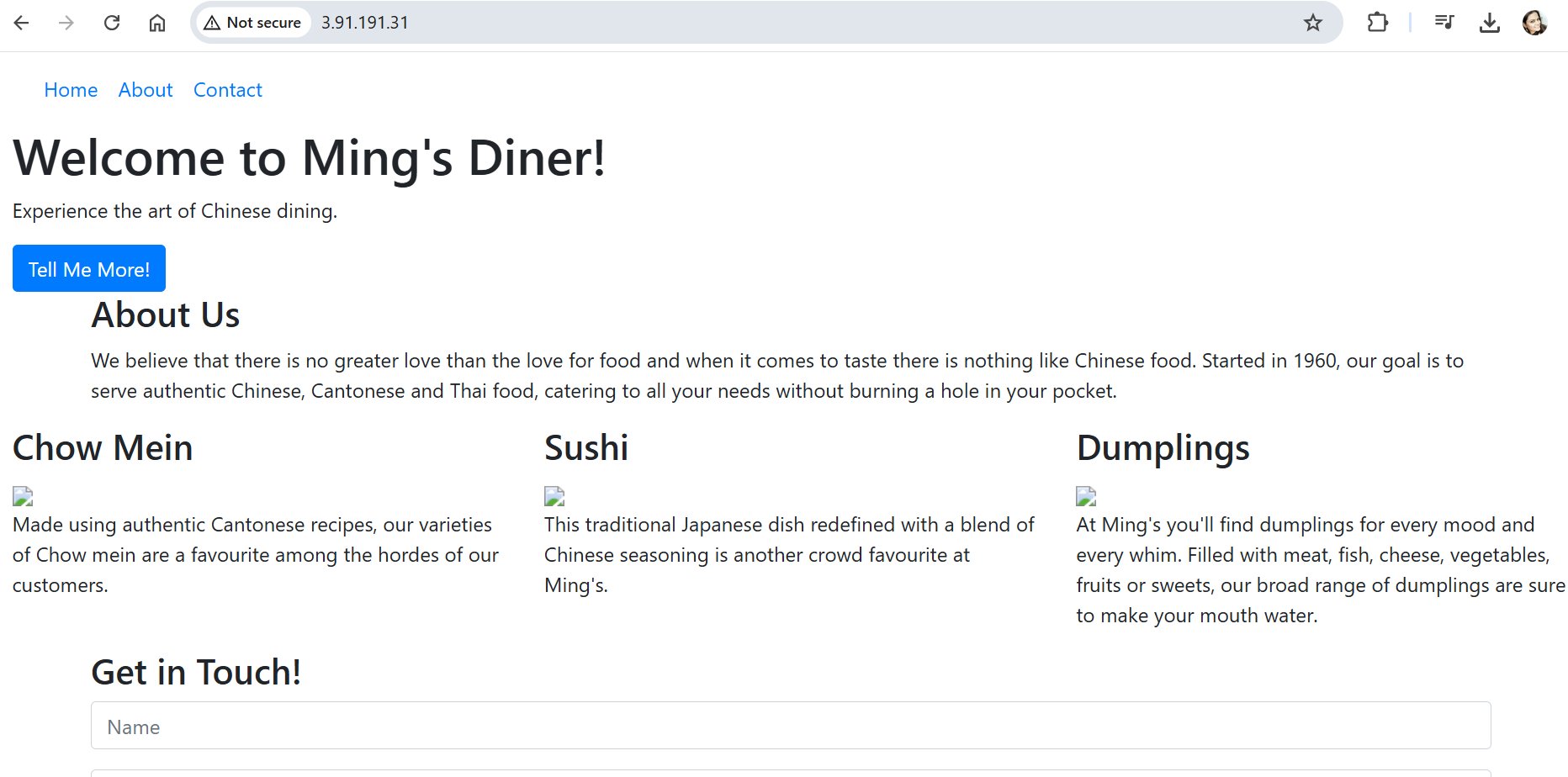
**Now go to browser in virtual machine and do the following settings   
**

**Now look for local host in the browser  
**

**Now the local host is accessible via internet.   
Now let’s try the same on our public address i.e. 3.91.191.31 of EC2 in our local host , that means website is successfully launched in our virtual machine.**

****

**Now , go to this pc folder then go to c drive and in inetpub’s wwwroot folder paste your website content, once this is done then access the ec2 via 3.91.191.31 on browser the result will be as show below:**

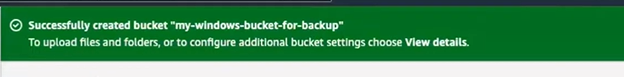
****

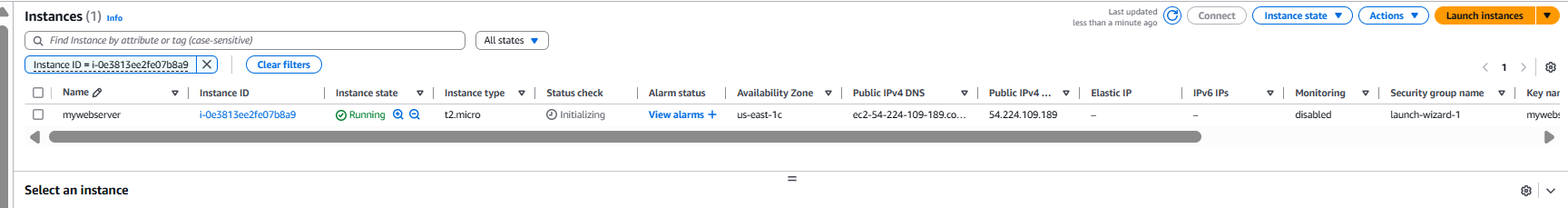
**Hence, website is successfully launched by a virtual machine.**

**Step 4 : Connect Linux VM to the Storage Service by connecting an AWS EC2 Instance to S3 bucket for storage.**

**41. Create a S3 bucket for storage:**

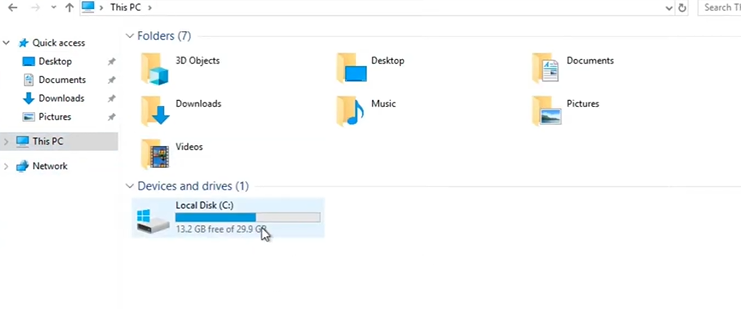
****

****

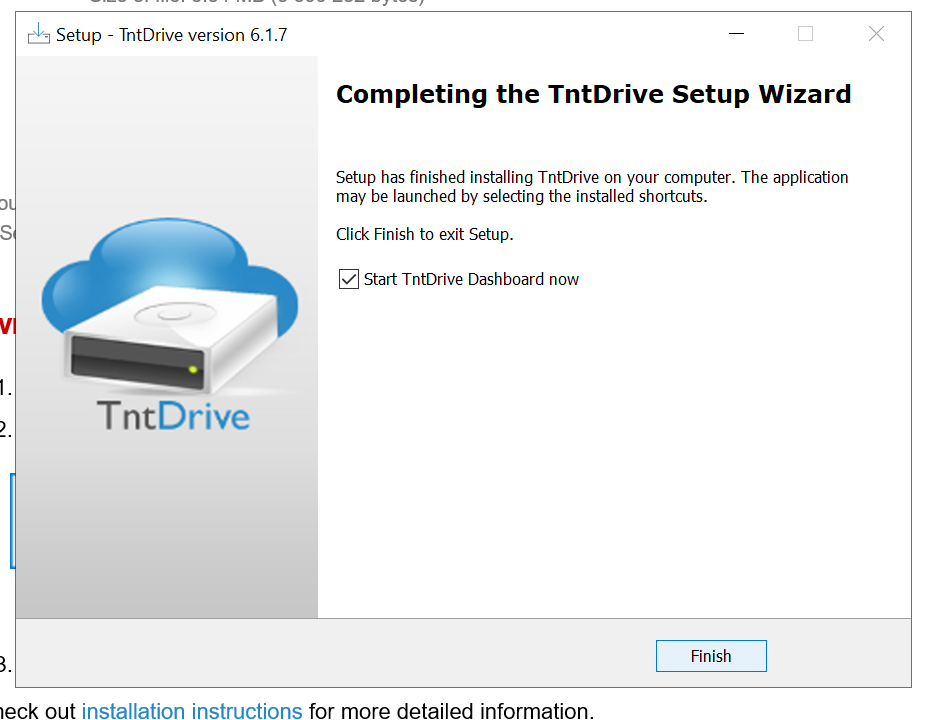
**4.2 Use the VM that we previously created:  
**

**4.3 As this is a windows VM so connect to the VM using RDP**

****

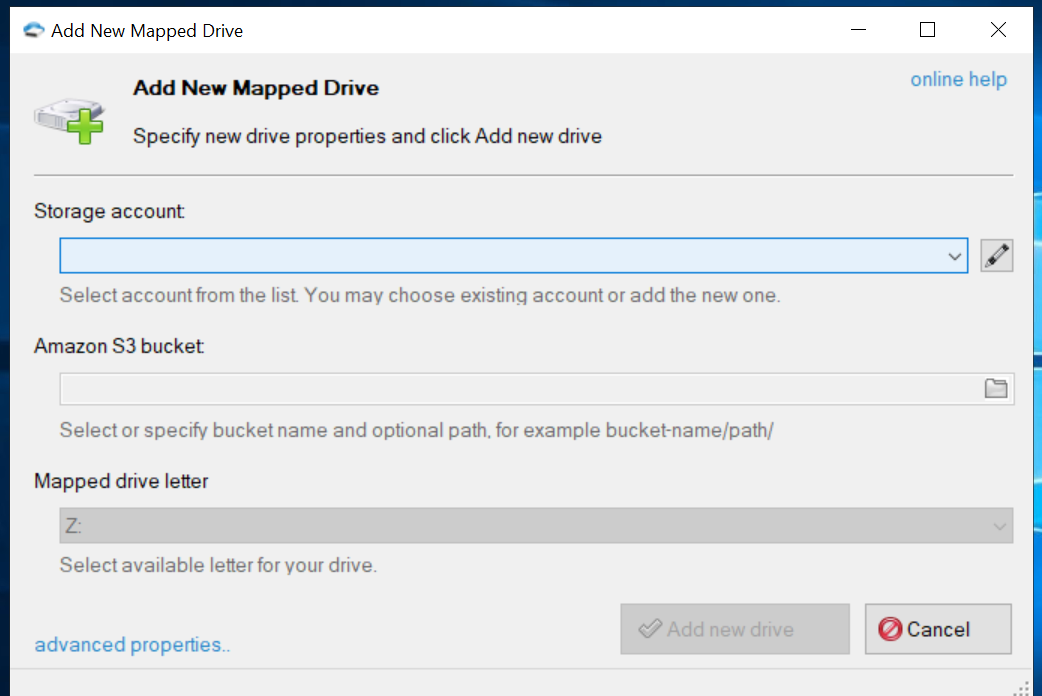
**4.4 Currently, there is only one drive as shown  
**

**4.5 To add the storage device to our ec2 VM , let’s go to google drive to download the TNT drive to support our storage s3 bucket into EC2**

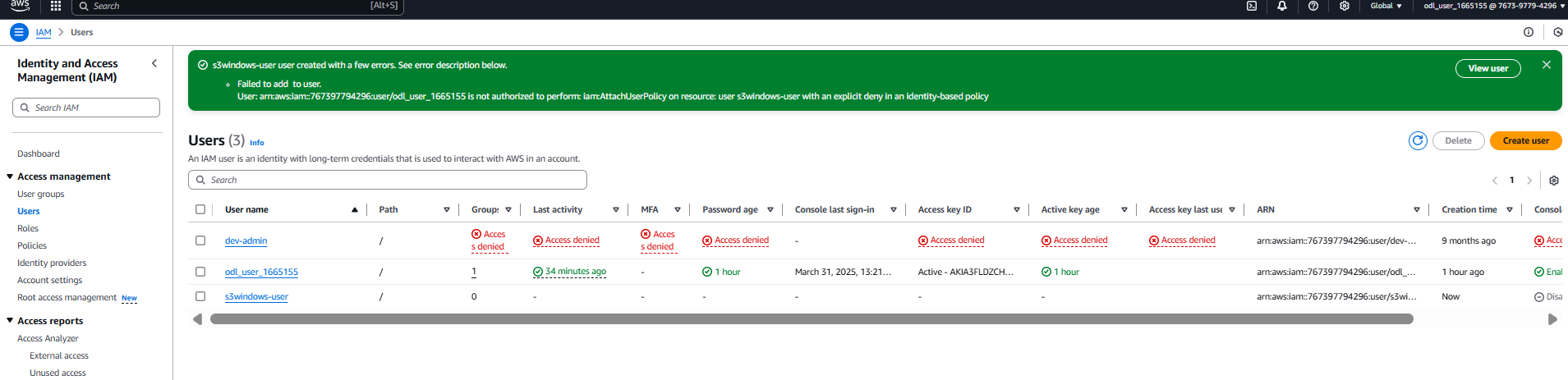
****

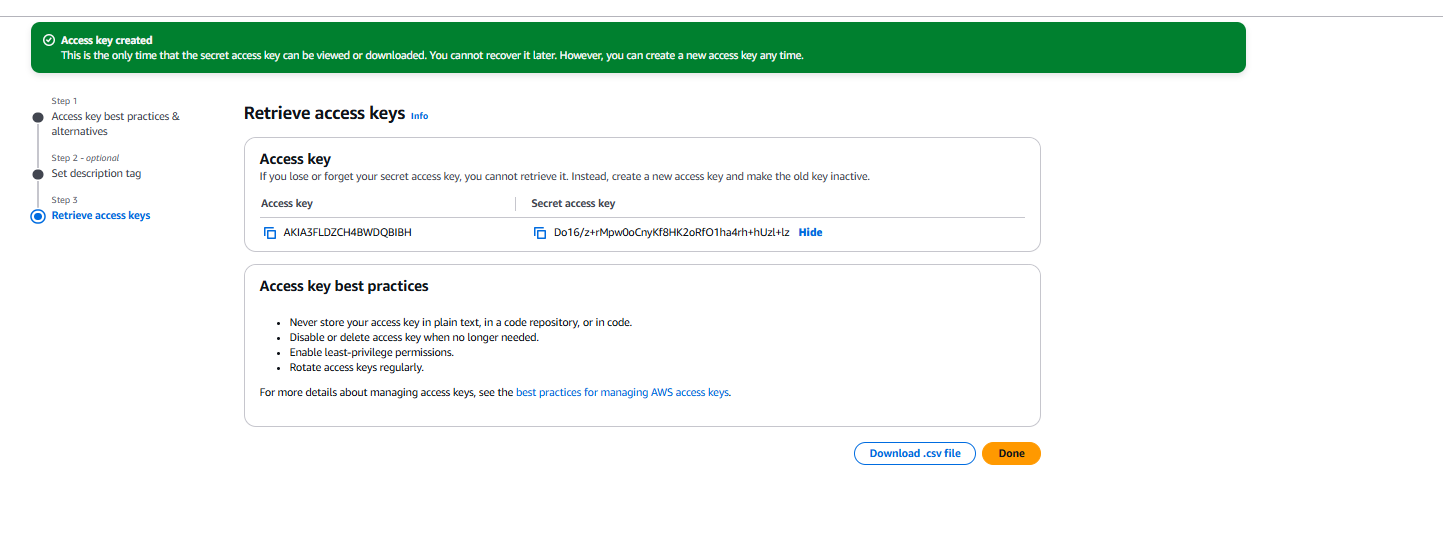
**Install the TNT drive from google.com click finish and run once installed  
**

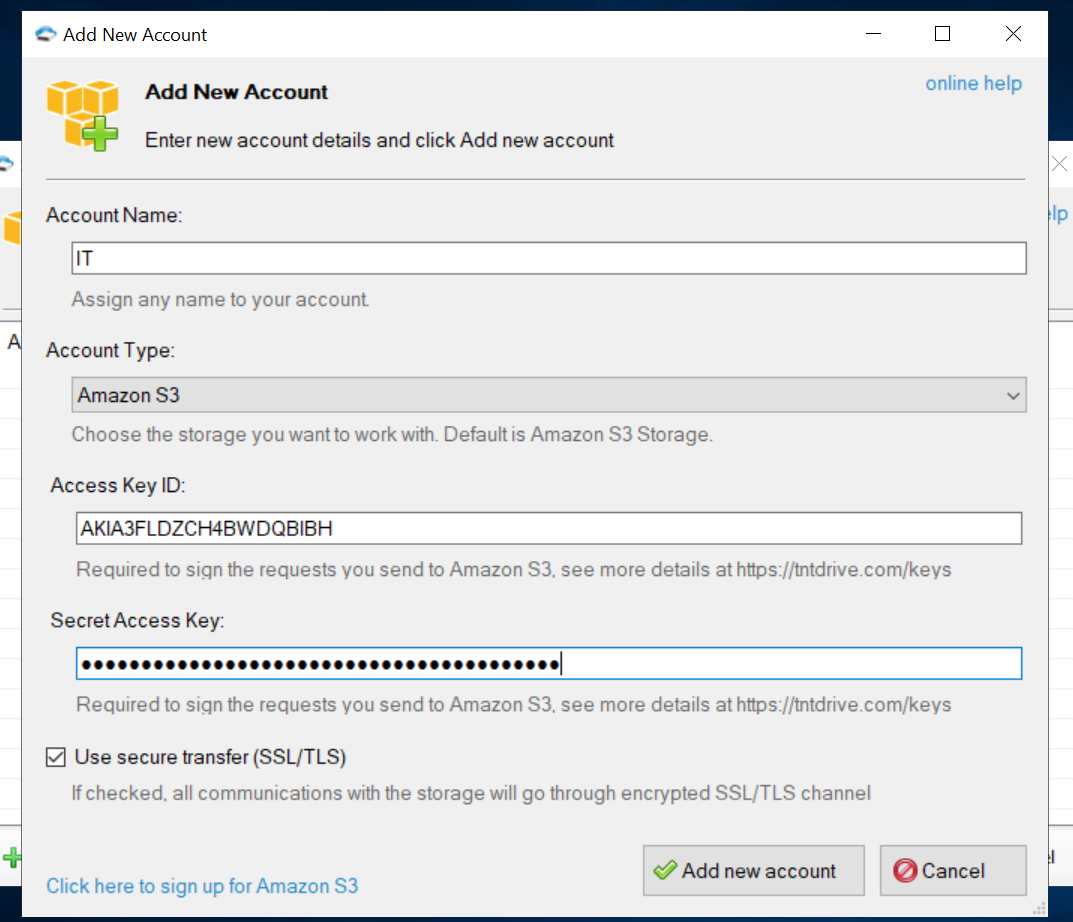
**TntDrive is finally installed on our VM**

**Now run the TNT program to map our drive as shown below  
**

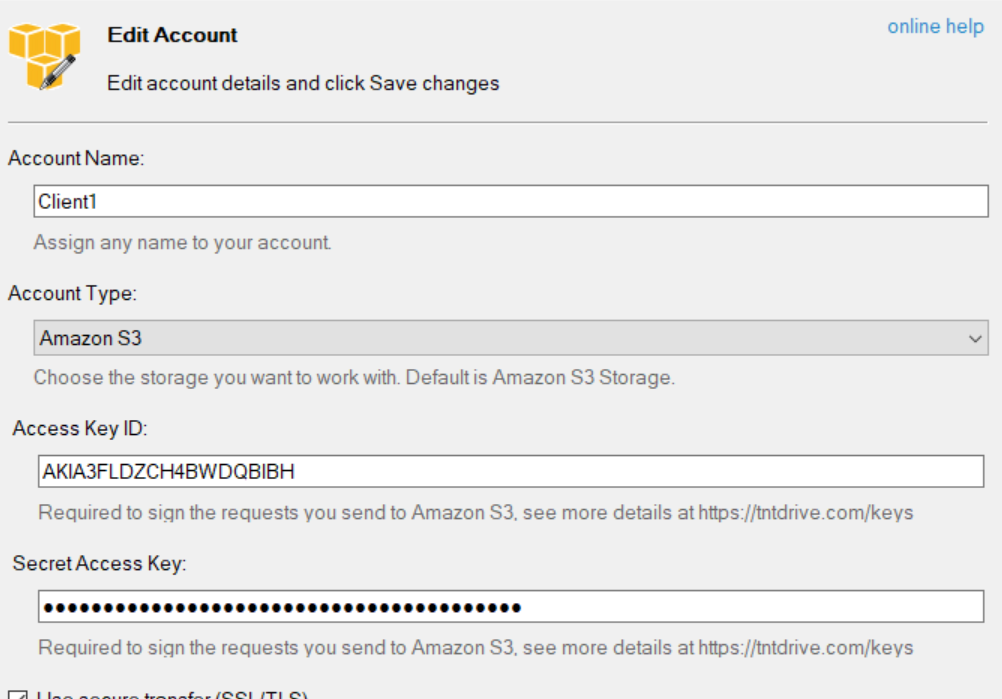
**Before mapping the device here, let’s create an IAM role to give a user fullaccess to s3 bucket we create  
Go to IAM console in AWS, Give user a name and attach S3full access  
**

**  
  
User is successfully created as shown below  
**

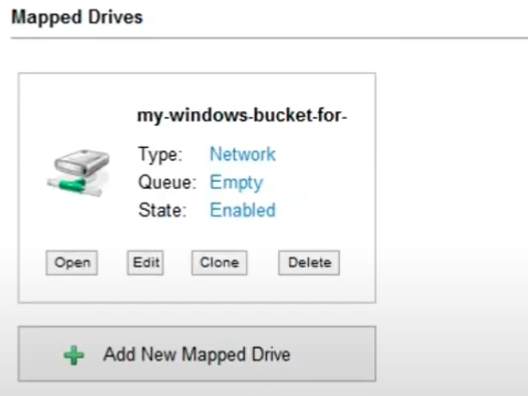
**After the IAM user is created, keep a not of the access key and secret key that will used while mounting s3 to our VM   
**

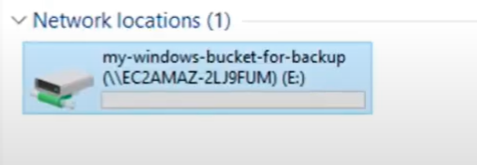
**Now go to the TNT drive to add a storage account to your Vm and share Access key and storage key to add the account as shown below  
**

**Now, mount the s3 as storage using TNT as shown below:**

****

**Now select the bucket we created and map it as E: drive as show below:**

****

**Now check this in you pc as E: drive as show below:  
**

**Hence, the bucket was successfully mounted on EC2 as storage device**