**Deploying an Online Movie Watching Application on Cloud**

Description

You are working in an online entertainment provider company. As you know about cloud computing, you are asked to deploy the company’s website on the cloud.

**Background of the problem statement:**

You work for Binge Watch Online, an online entertainment provider company.

You have created a website for the company and used a public cloud to deploy the website. After deploying it on the cloud, users are complaining about the reloading speed of the pages. The website is getting global traffic and static assets like pages that are served from a single server. You need to make sure that the traffic coming to the website from different parts of the world is load-balanced at the DNS level.

You can use either Azure or AWS platforms to design the solution using IaaS OR PaaS.

**You must use the following tools:**

* Azure: Azure App Service, CDN, DNS, Azure VM, Azure Traffic Manager

**You have been asked to:**

1. Suggest an appropriate solution so that your company can make use of the cloud while keeping the requirements mentioned above for your company in mind
2. Provide an approach to:

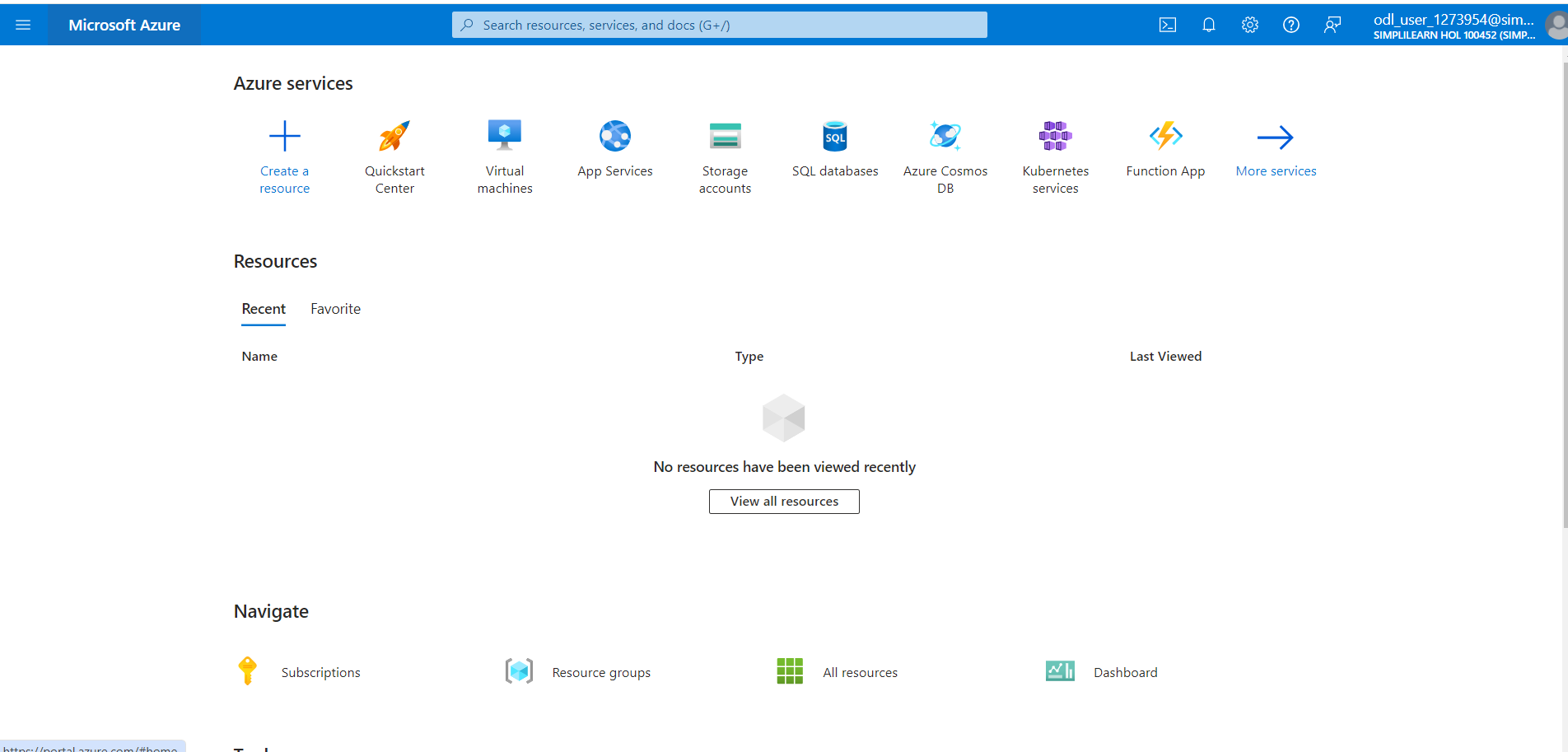
a. Govern all the resources being used for the development, testing, and production of the company’s website

b. Keep a separate track of the billing life cycle and cost management of all the services being used for hosting the company’s website on Cloud

1. Upload all static content of your website to the cloud
2. Create a CDN endpoint and configure it to serve the static files you have uploaded
3. Use storage service and upload files for your teammates to share
4. Connect a Windows or Linux VM to the Storage service

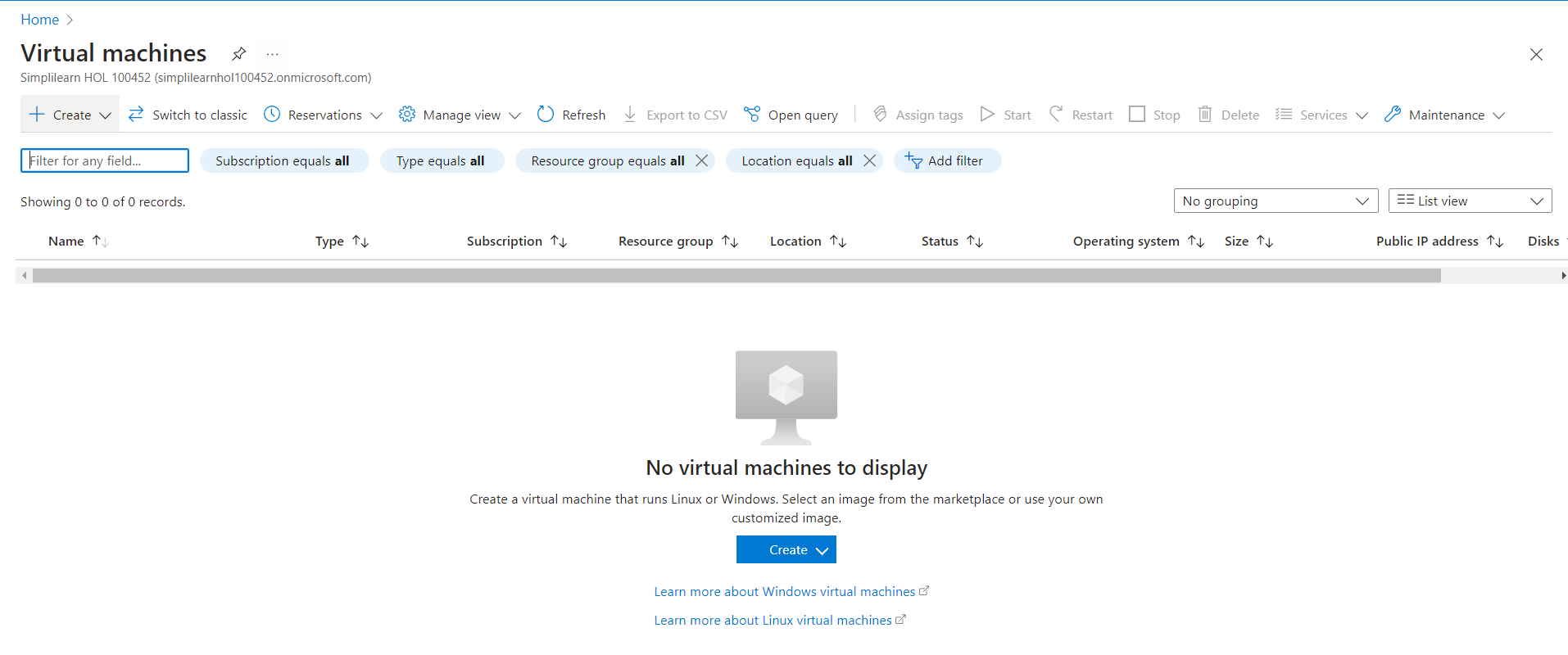
Project Approach 1

**Step 1:** Log into the Azure portal

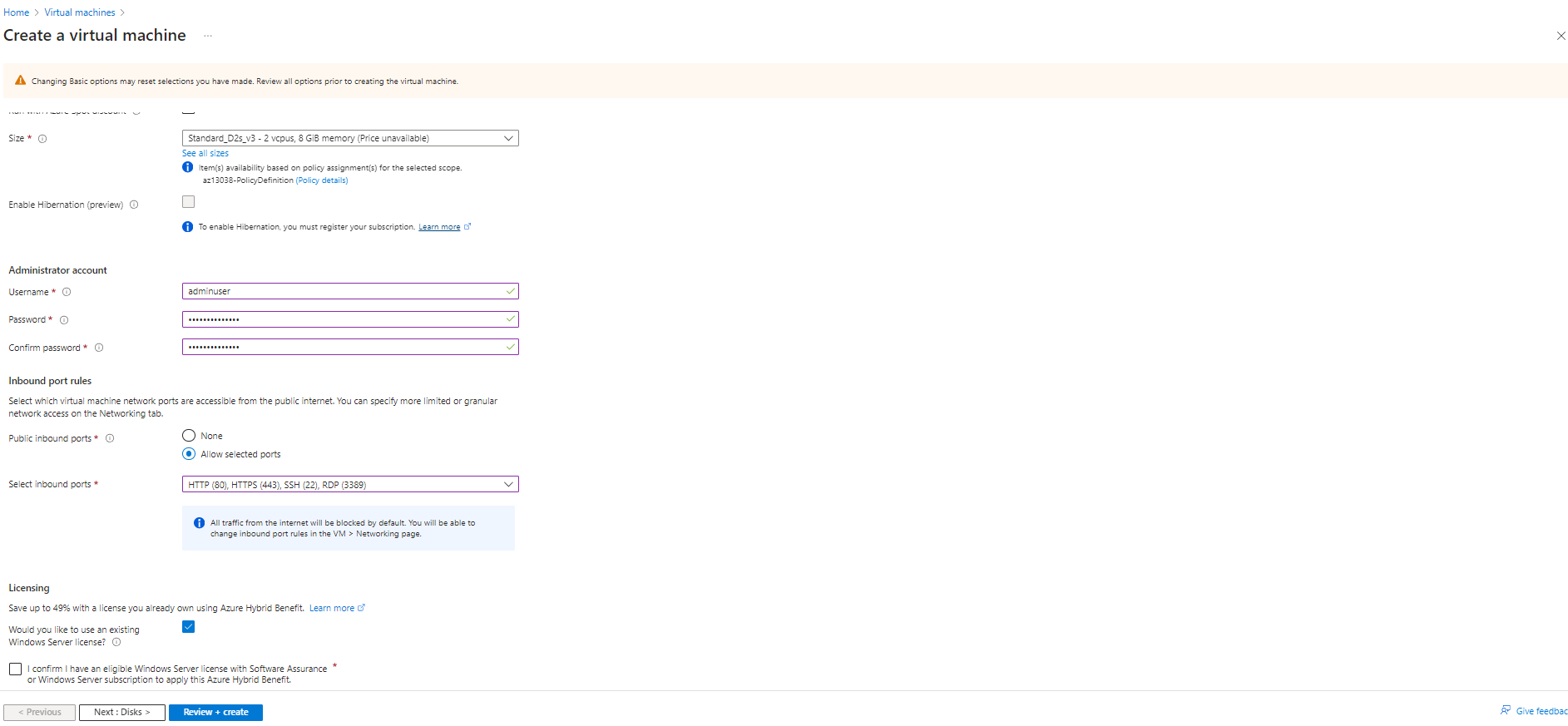


**Step 2:** To begin, create an Azure VM.

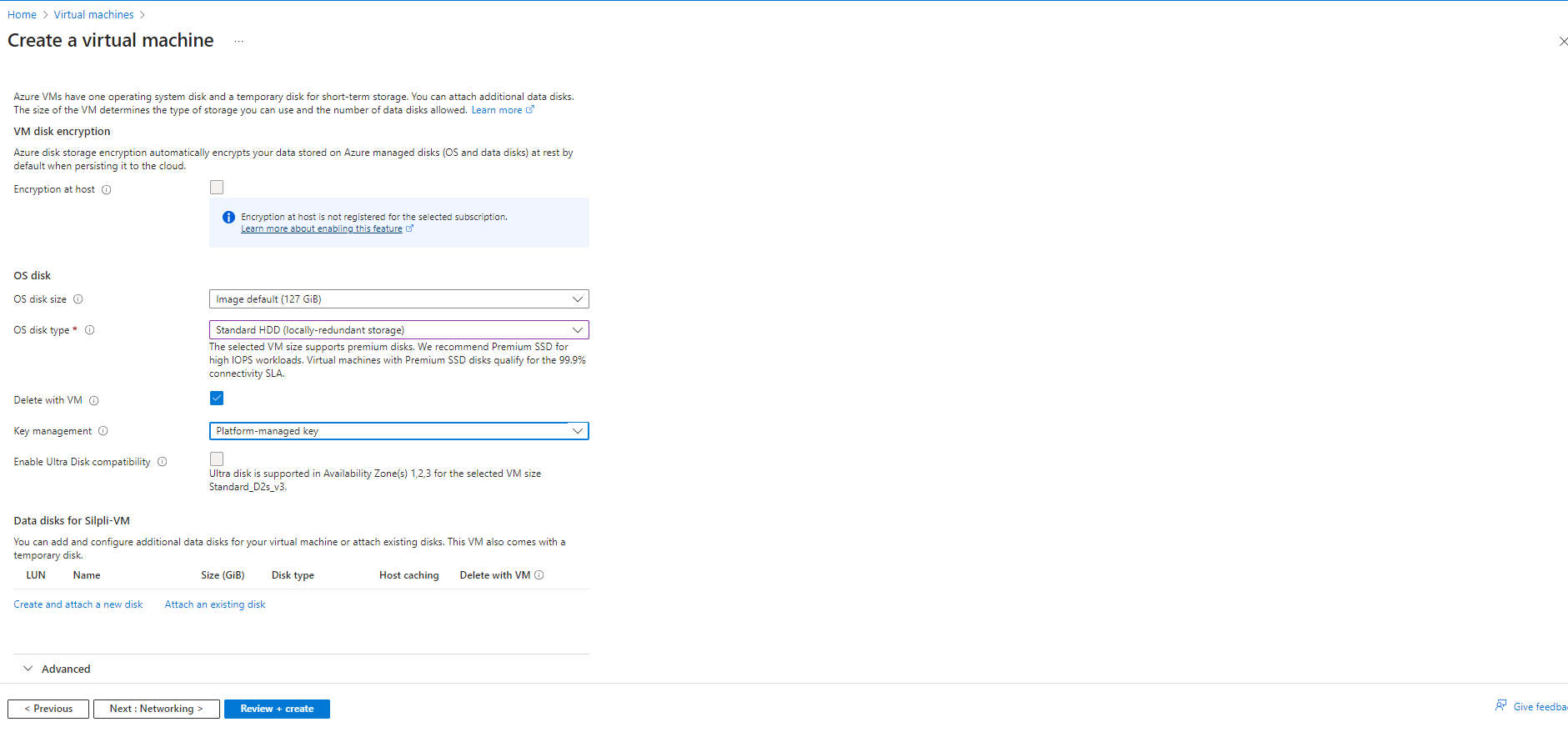
Step 2.1: Search for Virtual Machines and click on **Add**



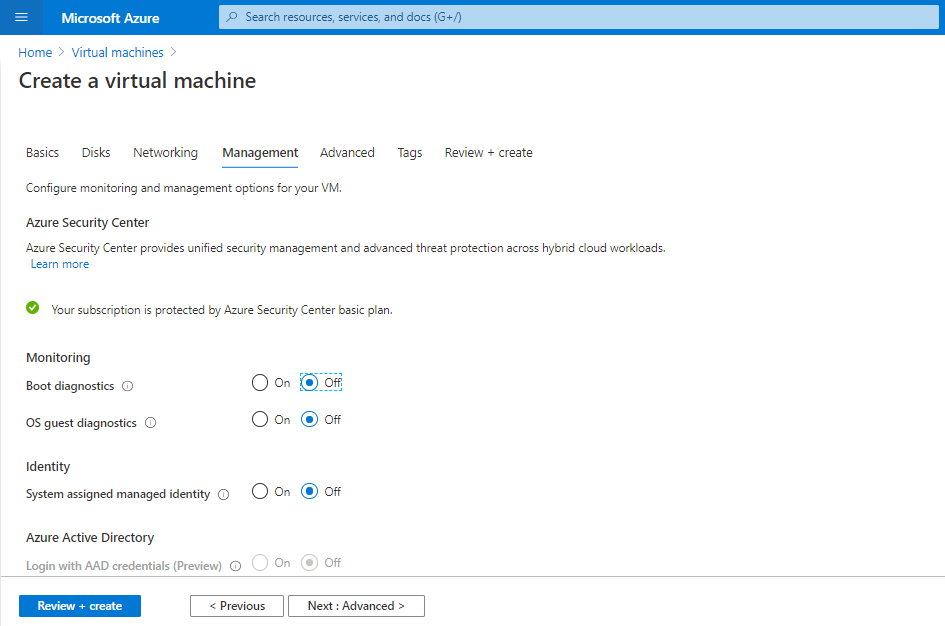
Step 2.2: Provide basic information about VM



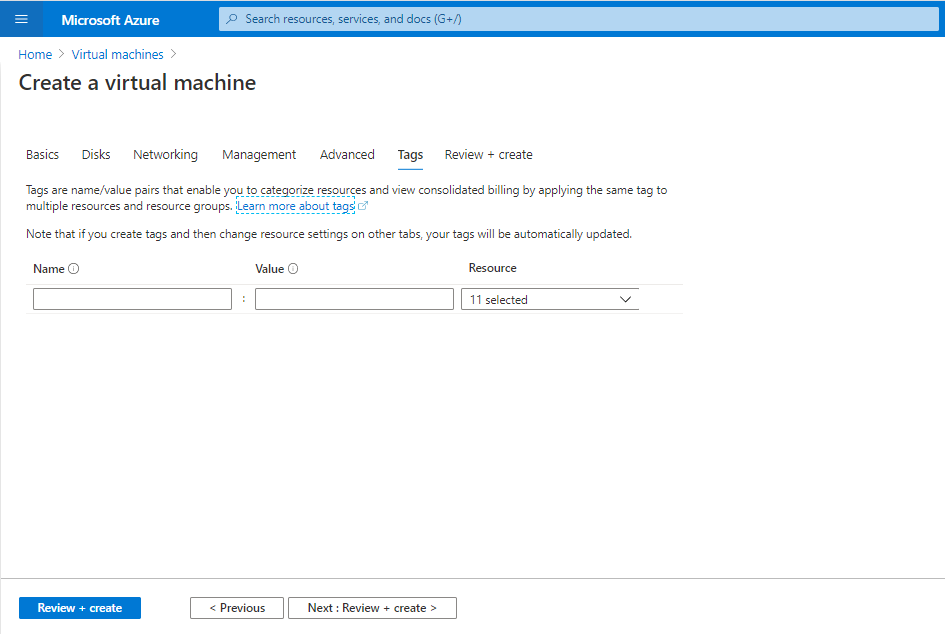
Step 3.3: In the Disks section, provide the required information



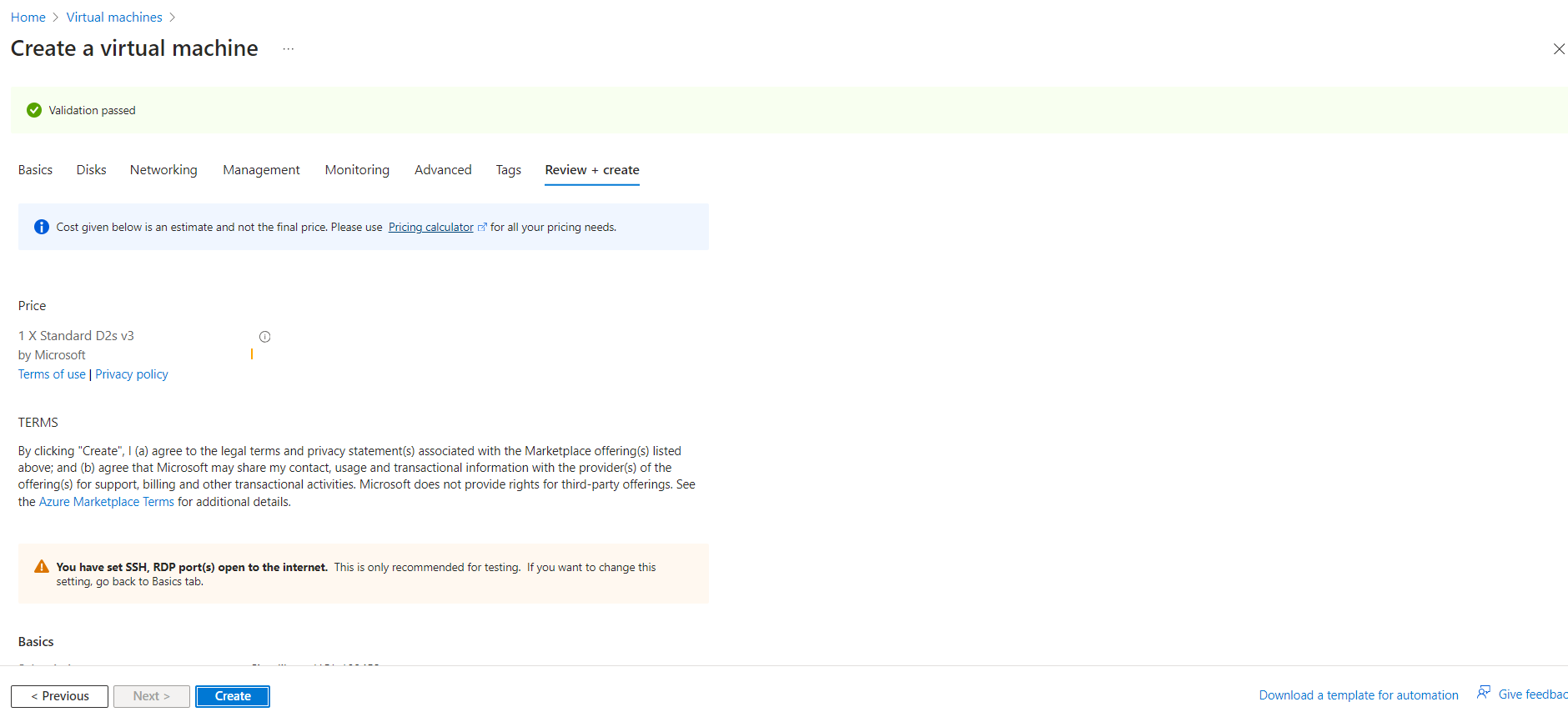
Step 3.4: In the Management section, turn off the Boot diagnostics



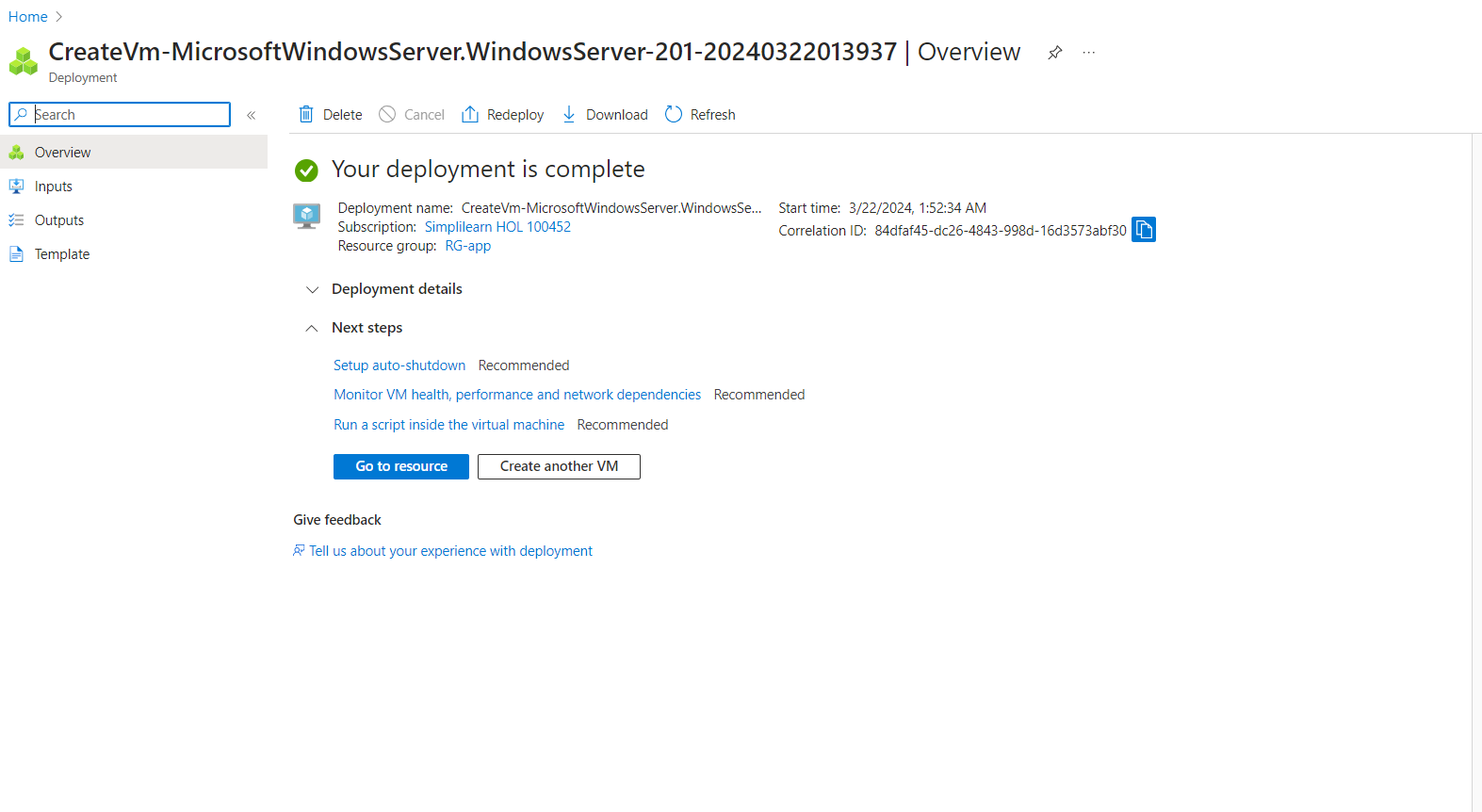
Step 3.4: Click on **Review and Create**



Step 3.5: Click on **Create**

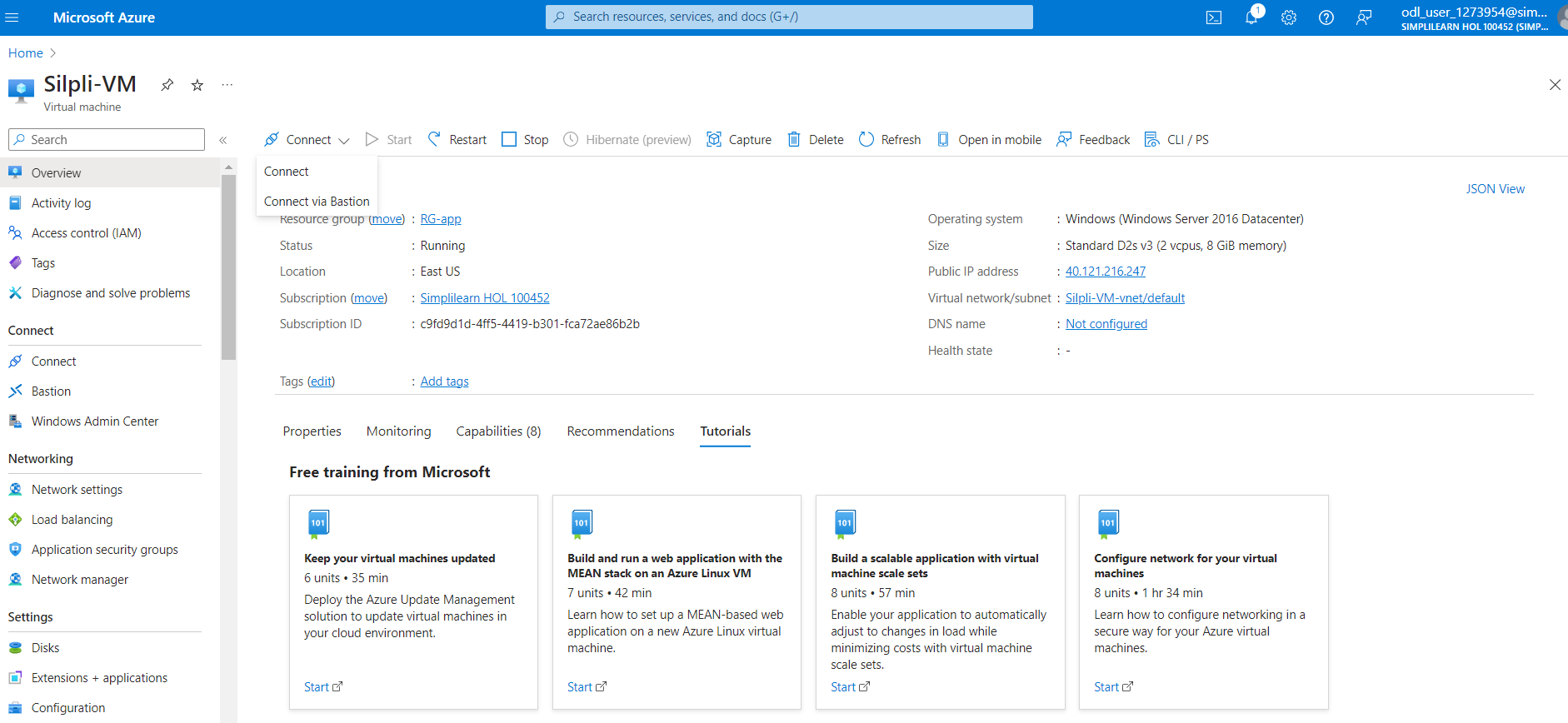


Step 3.6: Your VM will get deployed.



**Step 4:** Log into Azure VM and spin up a web server of your choice on port 80.

Step 4.1: Click on **Connect** and select **RDP**

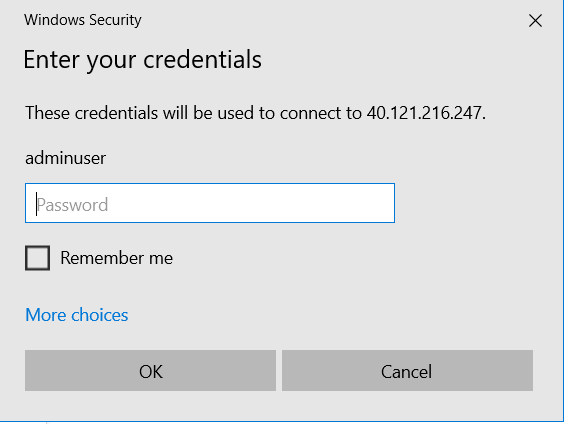


Step 4.2: Click on **Download RDP file.** It will download the created VM in your system.

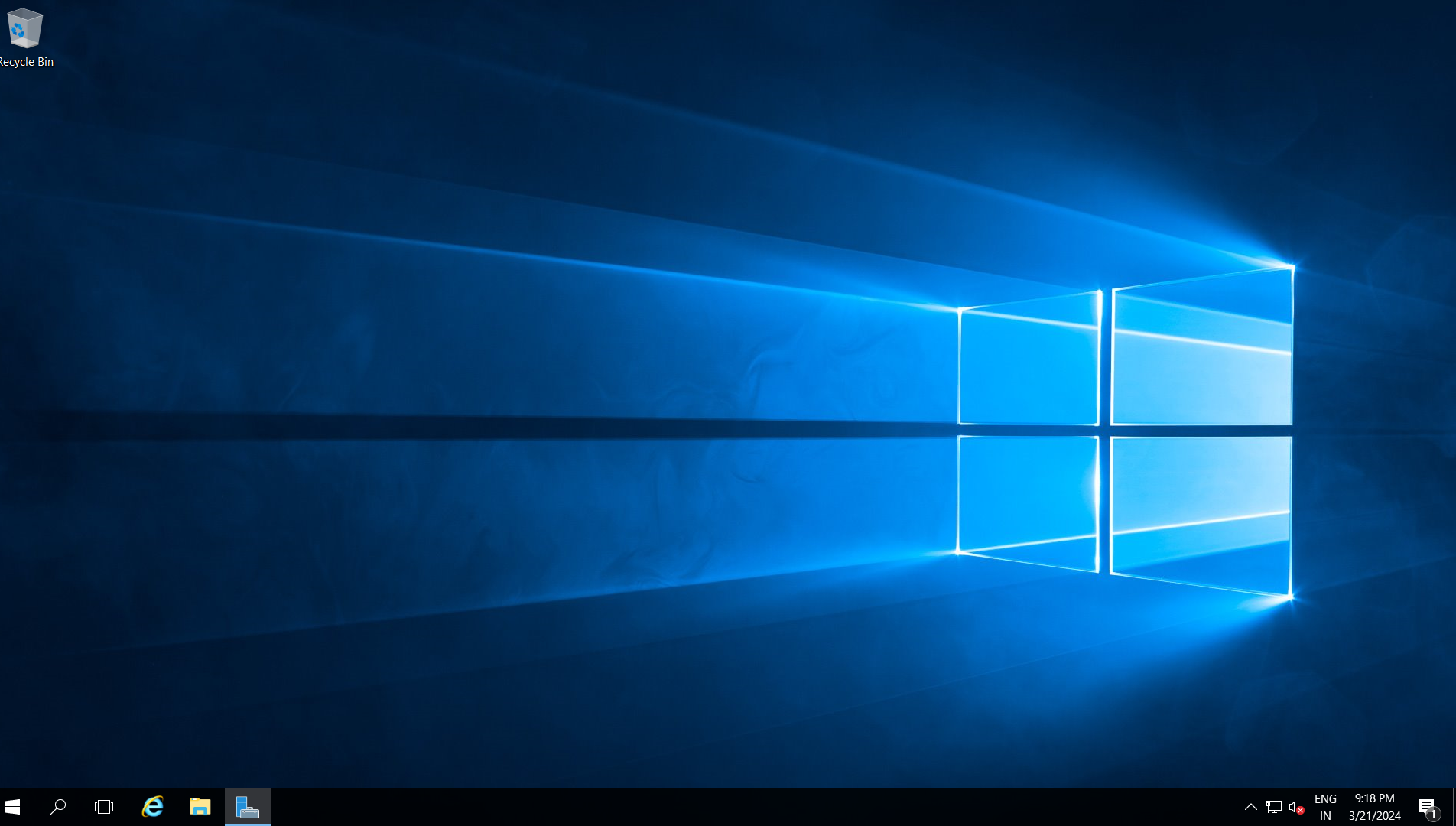
Step 4.3: Open the downloaded VM. Click on **Connect**



Step 6.4: Enter the credentials to login into VM and click on **OK**

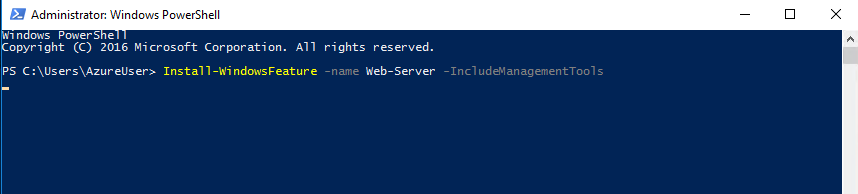


Step 4.5: This will open the VM in your system.

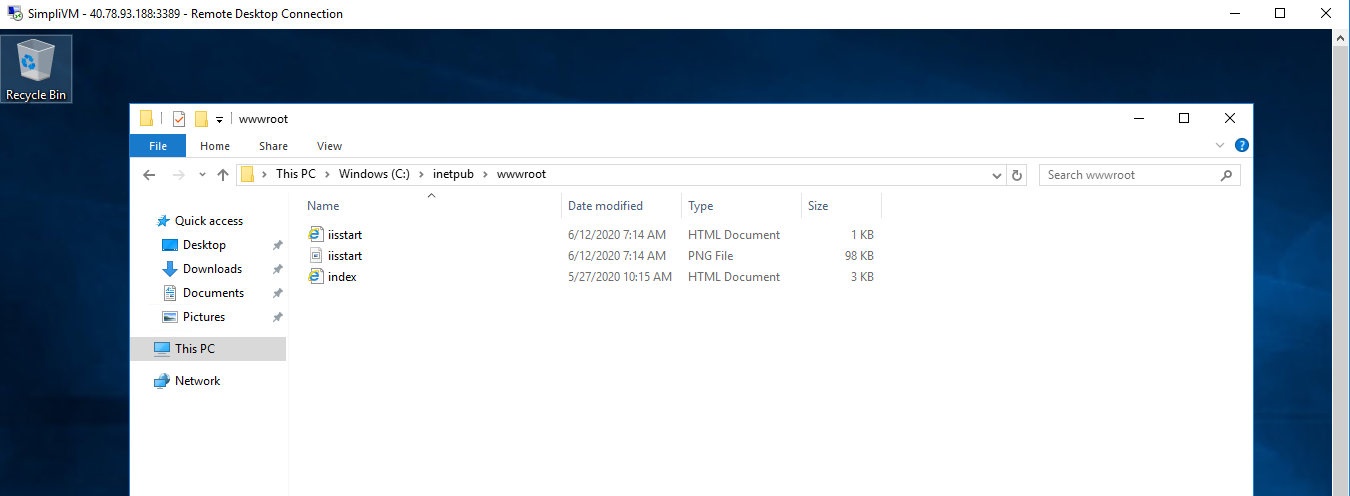


Step 4.6: Open the PowerShell and type the following command:

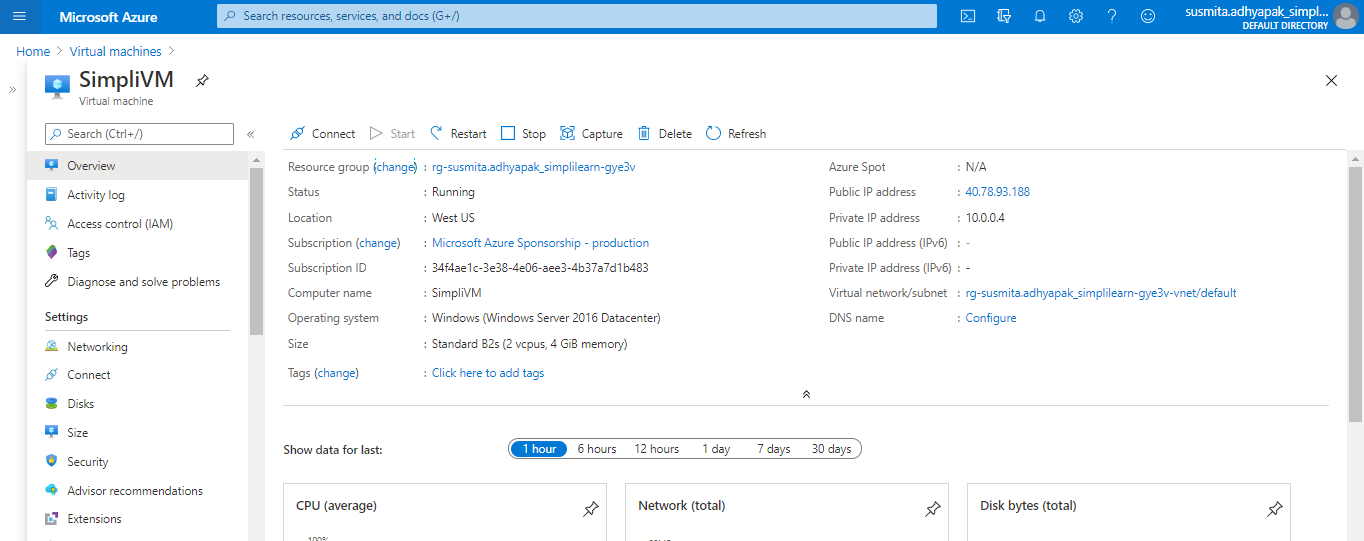
*Install-WindowsFeature -name Web-Server -IncludeManagementTools*

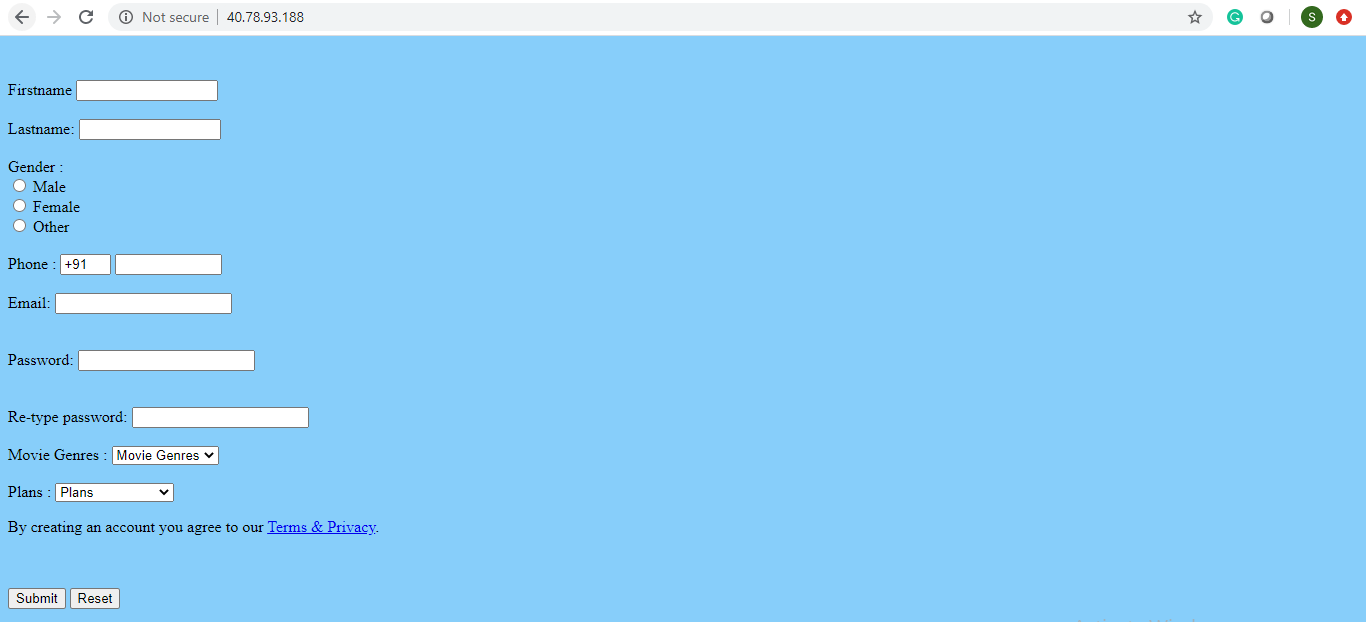


Step 4.7: The above command will create an inetpub folder in your C drive. Go to C:\inetpub\wwwroot and copy-paste the index.html file of your application which you want to deploy here:

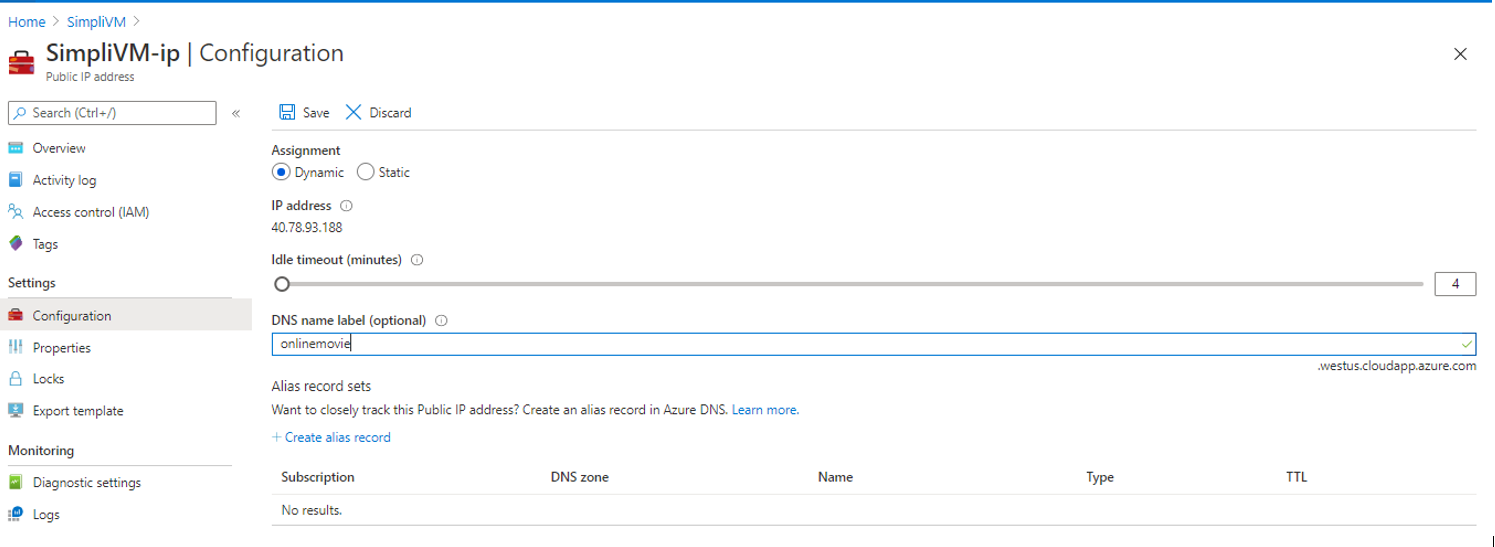


Step 4.8: Copy and paste the public IP address of the created VM in the browser and you will get the output

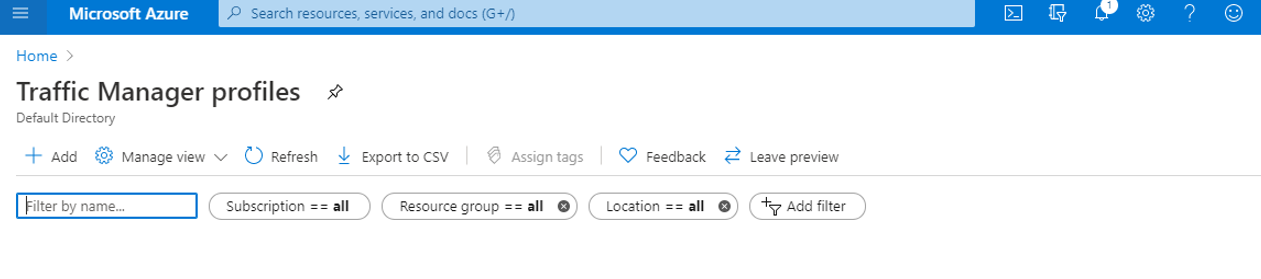




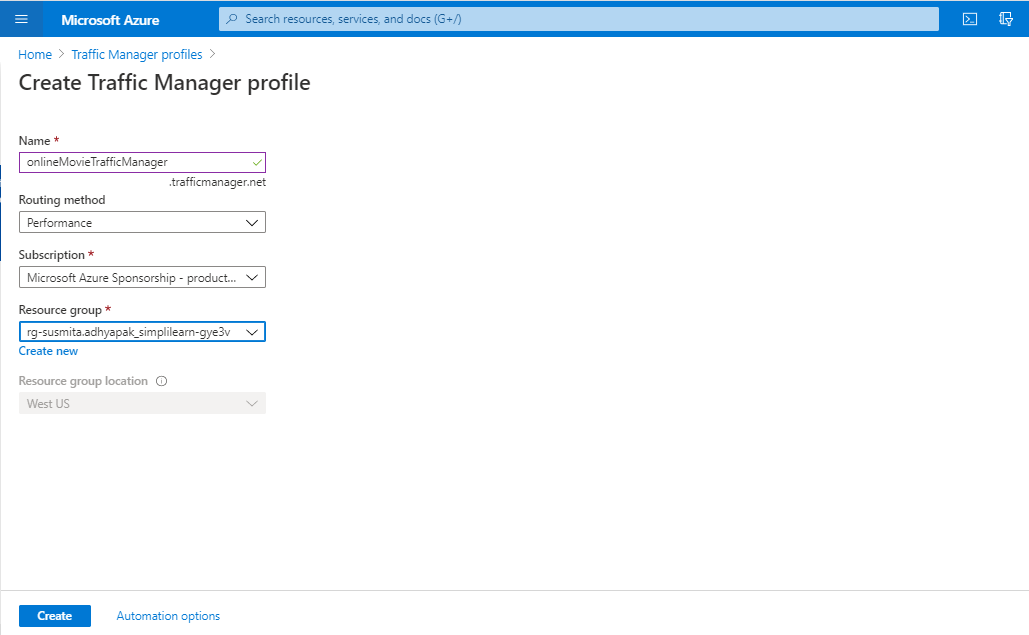
Step 4.9: Click on the Public IP address of the created VM



Step 5.1: In the search window, search for Traffic Manager Profile. Click on **Add** to create a new traffic manager profile

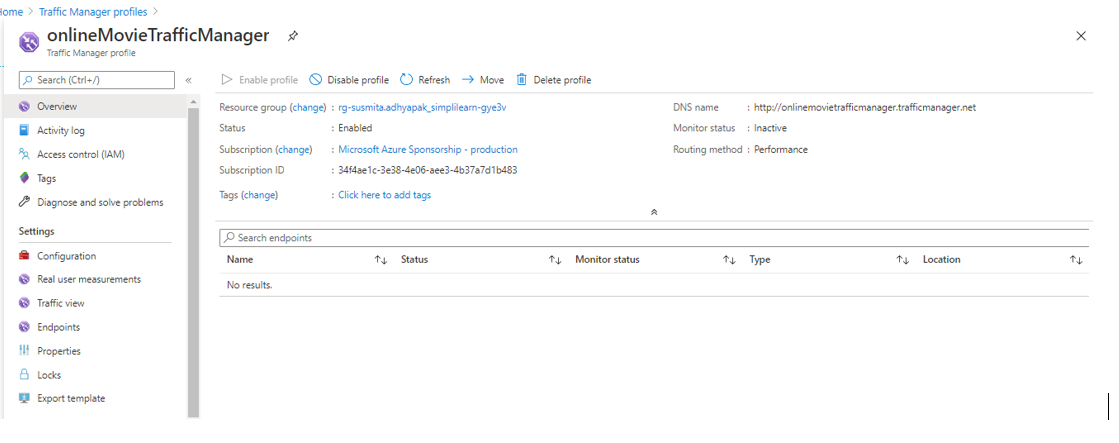


Step 5.2: Provide the required information and click on **Create** to create the Traffic Manager Profile

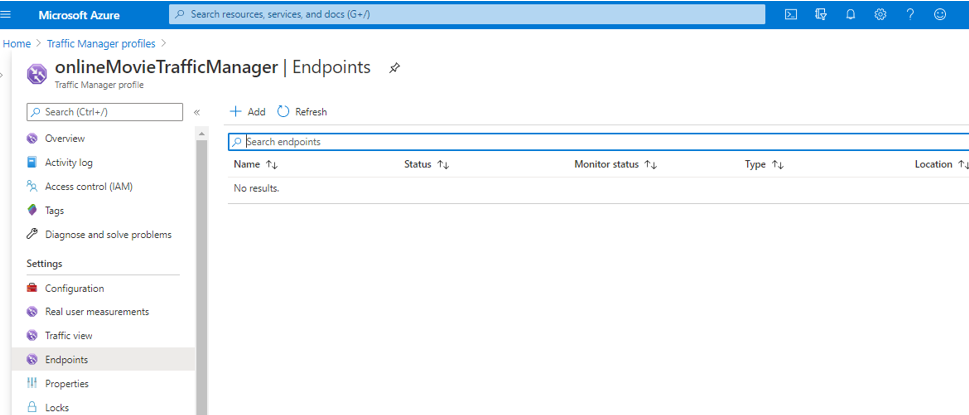


**Step 5.3:** Create endpoints in the traffic manager corresponding to public IP of each virtual machine that you have created.

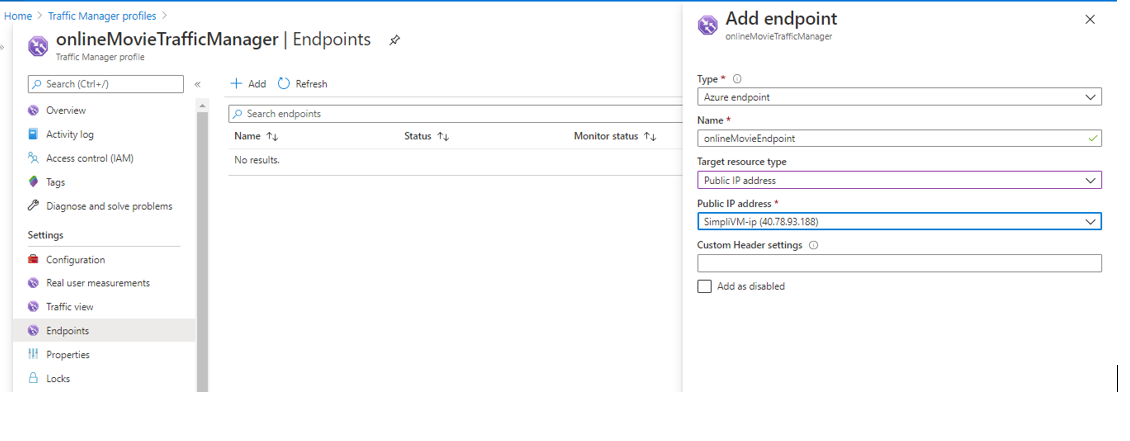
Step 5.4: Go to the created Traffic Manager Profile



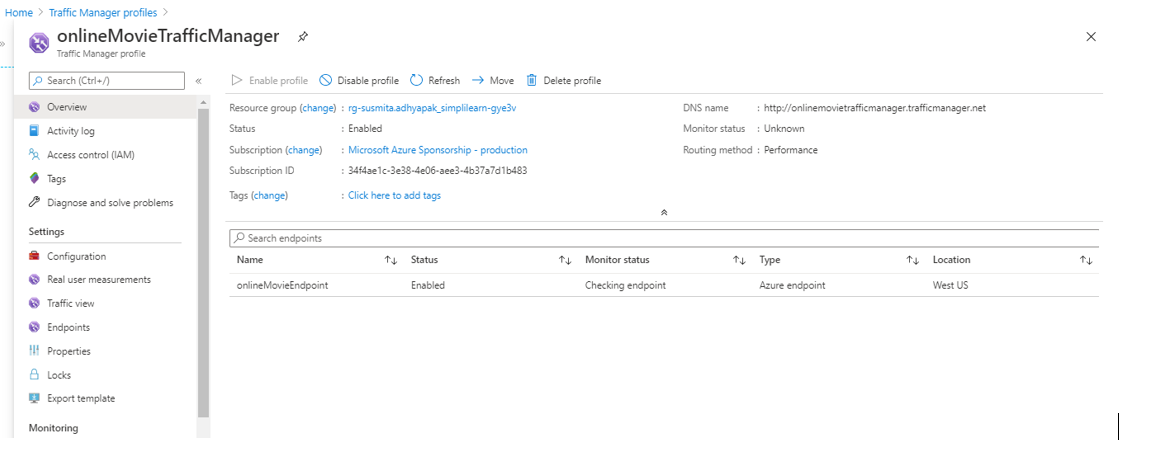
Step 5.5: Click on **Endpoints**. Click on **Add** to add new endpoints

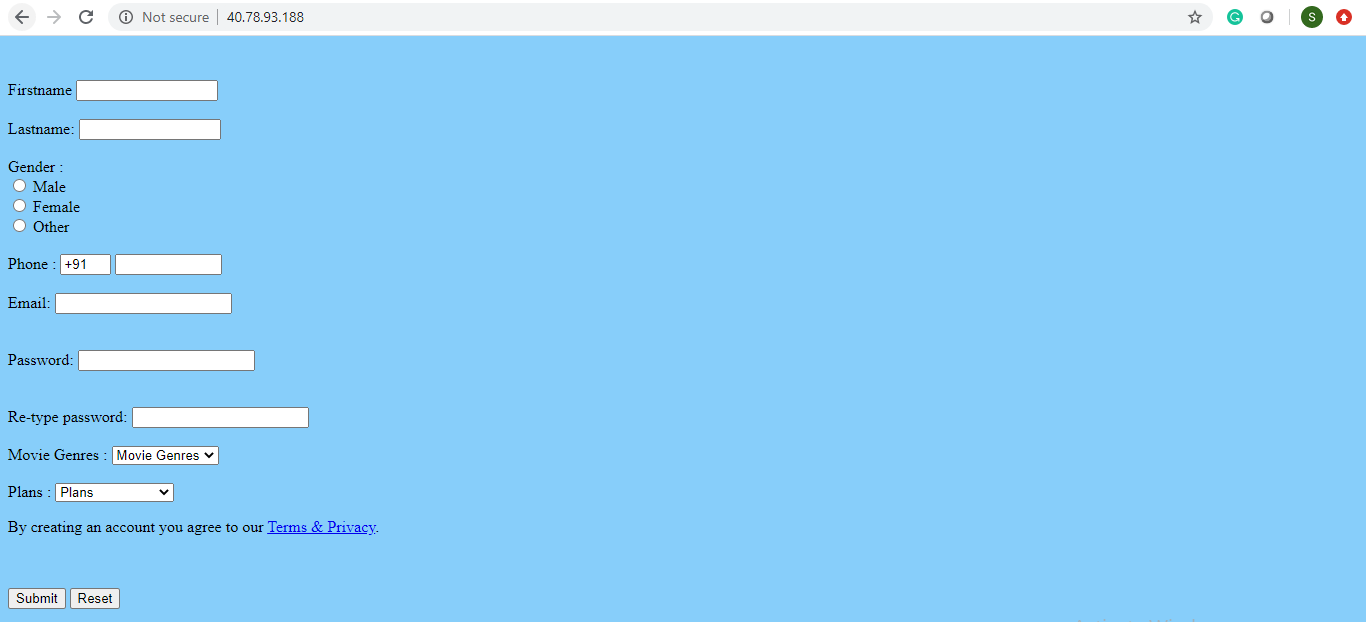


Step 5.6: Provide the required information and click on **Add**



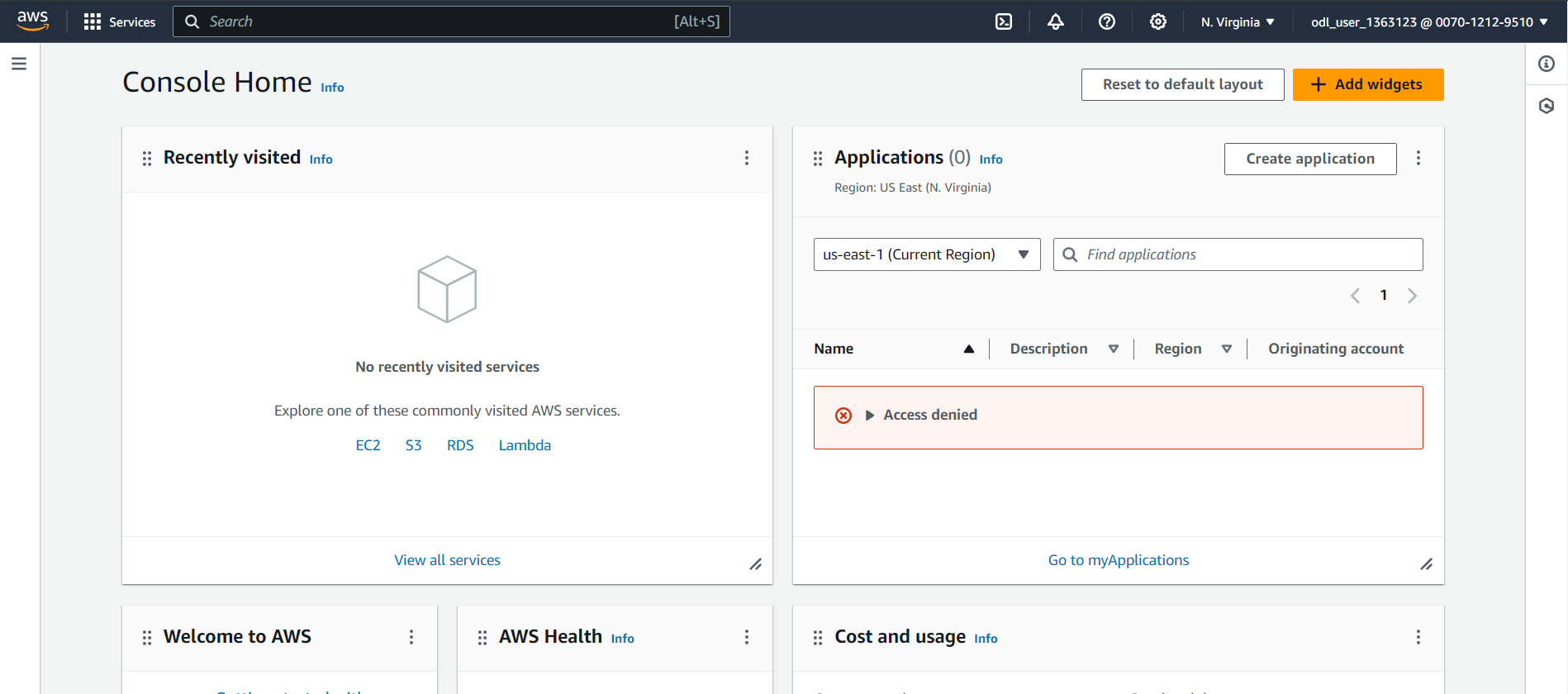
Step 5.7: Copy-paste the DNS name link in the web browser and you’ll get the output





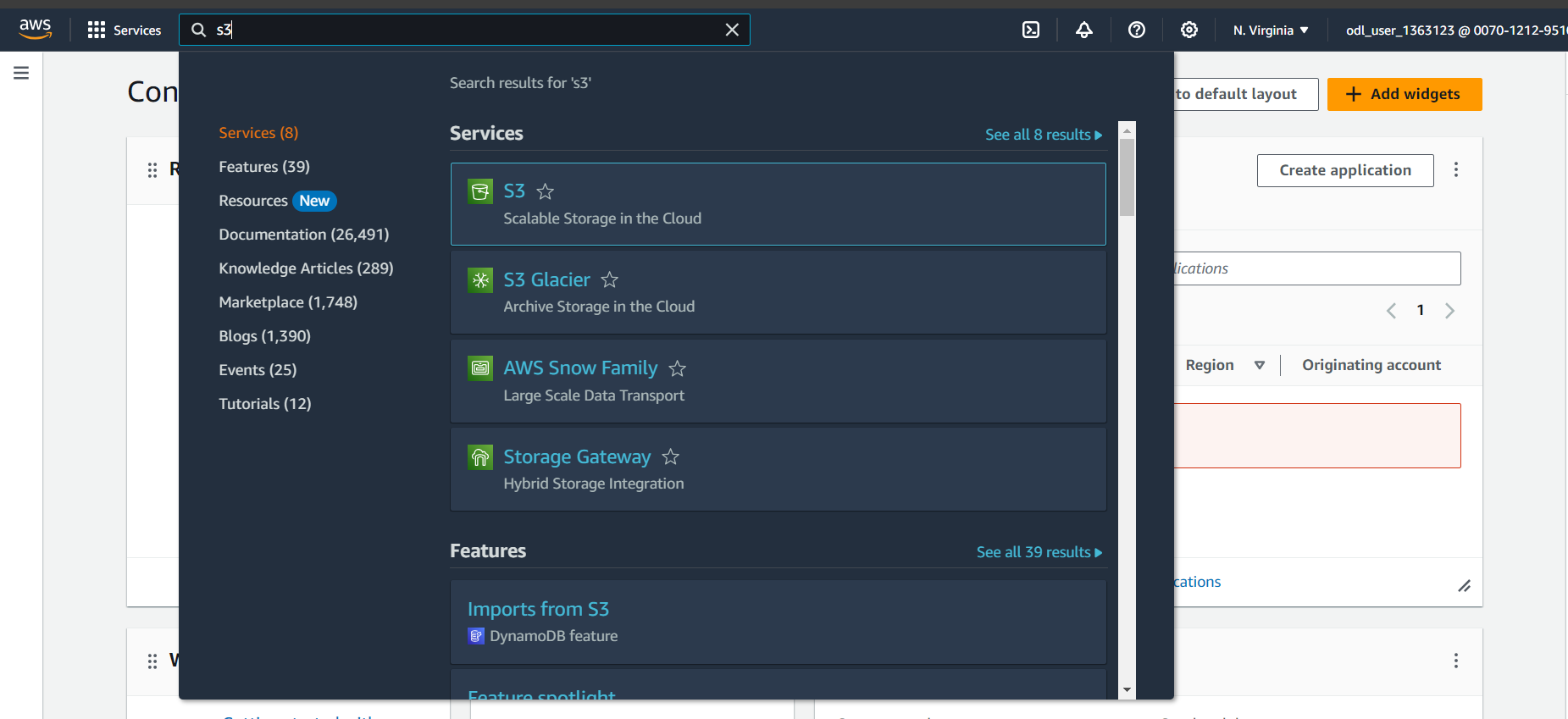
AWS

**Step 1:** Log into the AWS console



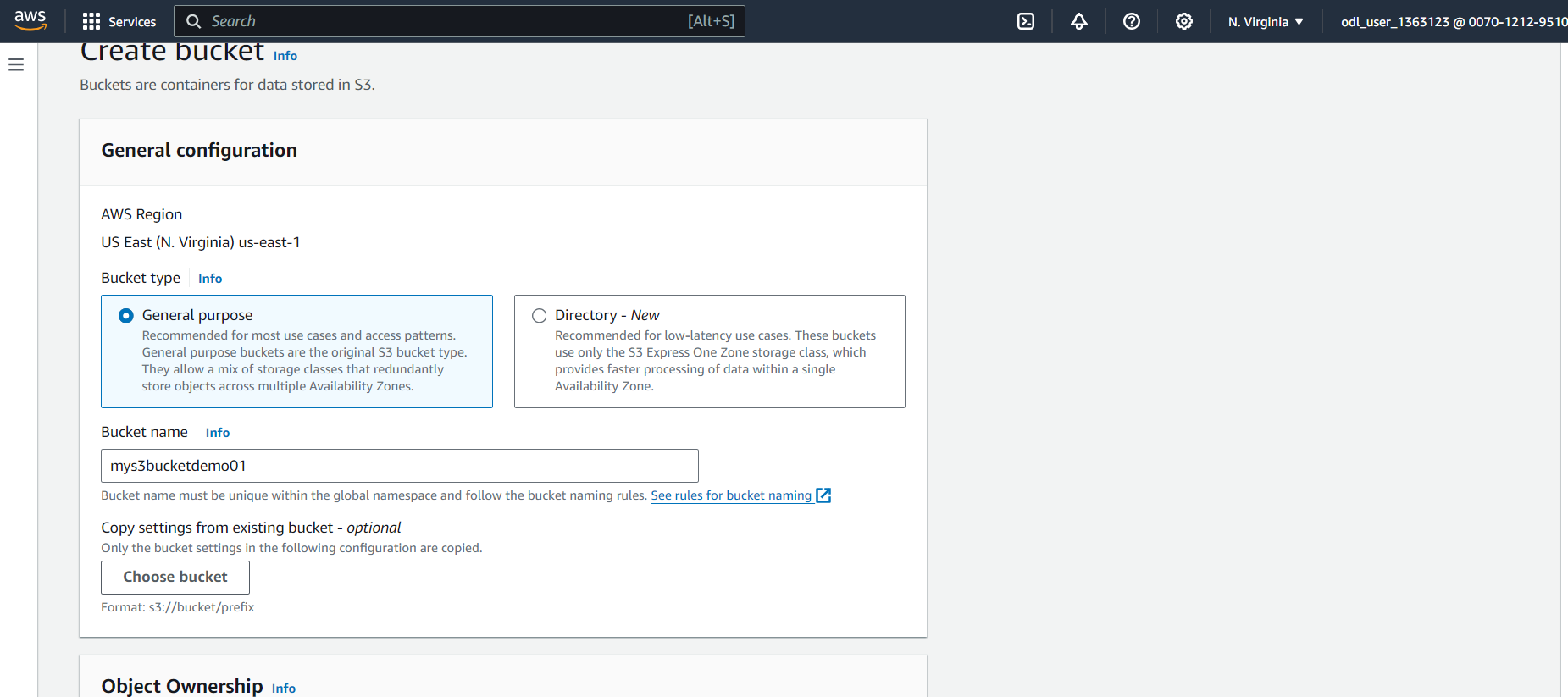
**Step 2:** Create an S3 bucket

Step 2.1: In the search window, search for S3 service

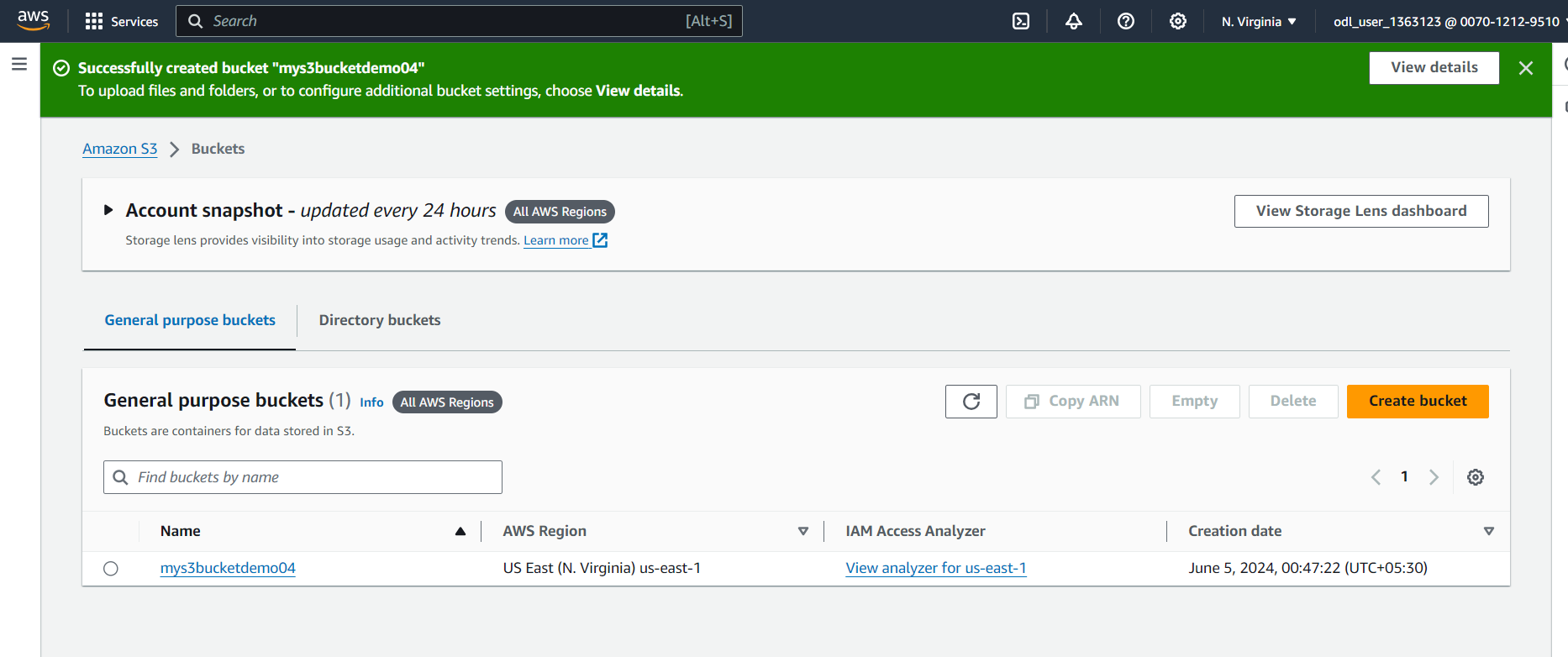


Step 2.2: Click on **Create a bucket**

Step 2.3: Provide the bucket name, select the region, and click on **Next**



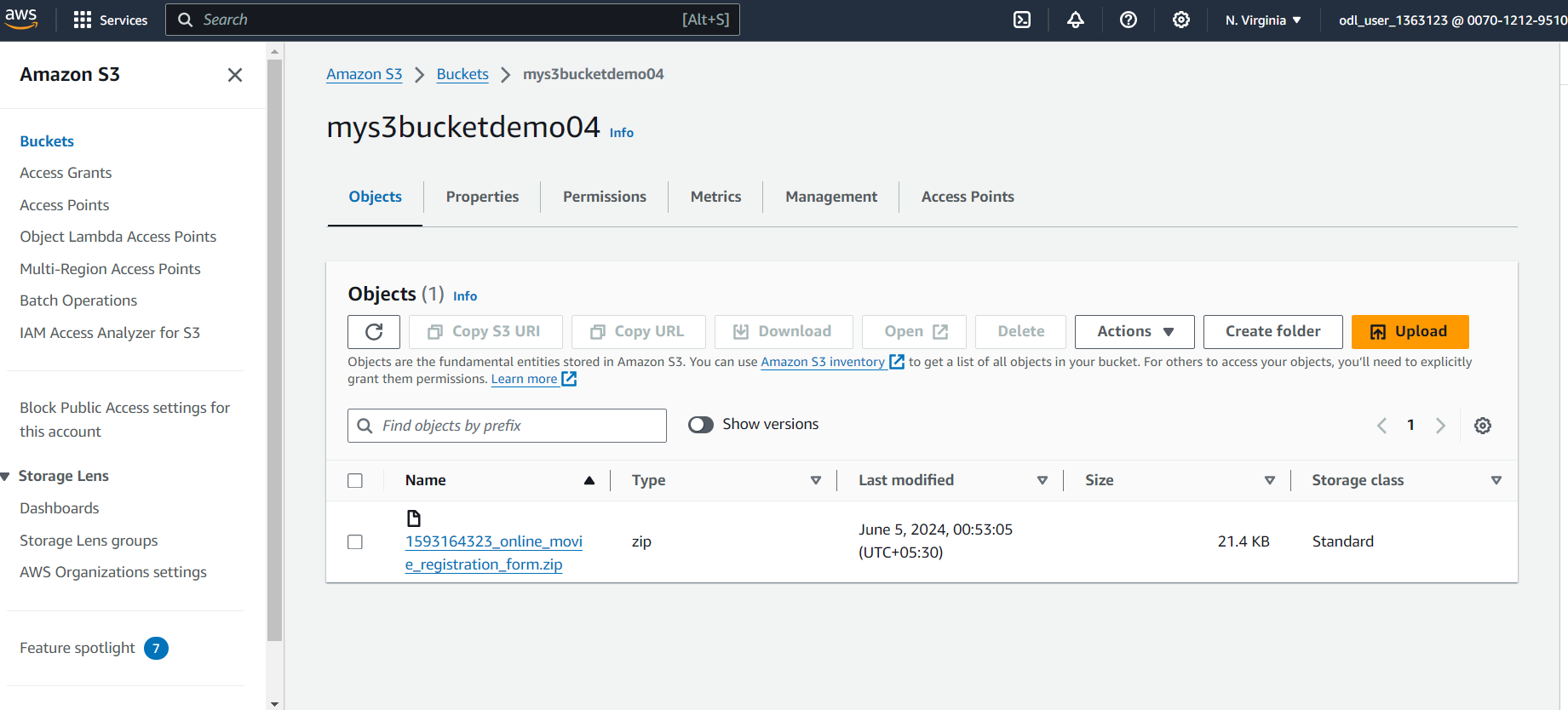
Step 2.4: The created bucket will be visible in the portal



**Step 2.5:** In the properties of the S3 bucket, configure the S3 bucket to enable Static website hosting

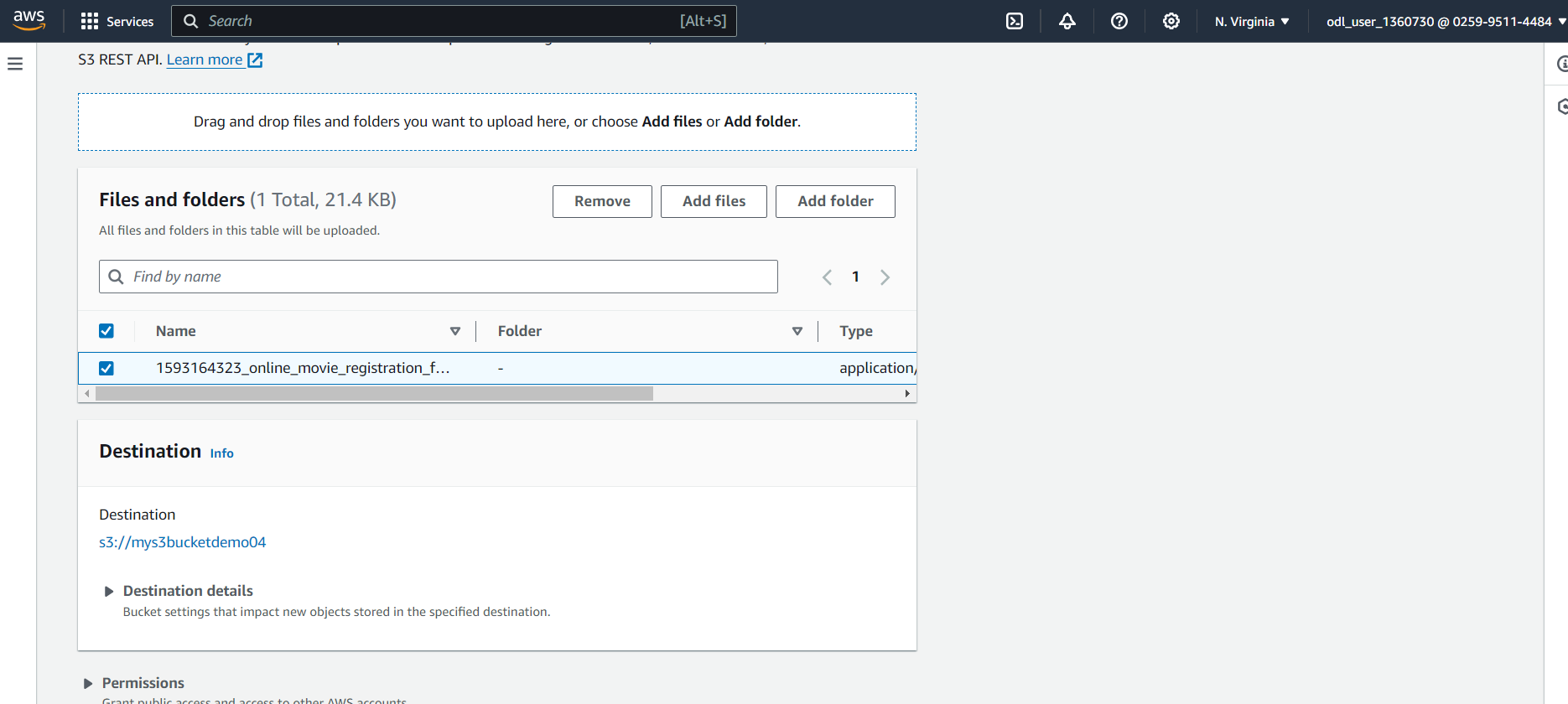
Step 2.6: Click on the created bucket

Step 2.7: Go to **Properties**



**Step 2.8:** Upload web app files to the S3 bucket.

Step 2.9: Go to the Overview tab of the created bucket and click on **Upload**



**Step 3:** Configure permissions in S3 and add the below bucket policy to give read-only access to the static web app endpoint

Step 3.1: Go to the **Permissions**, click on **Bucket policy**, add the following code there, and click on **Save**

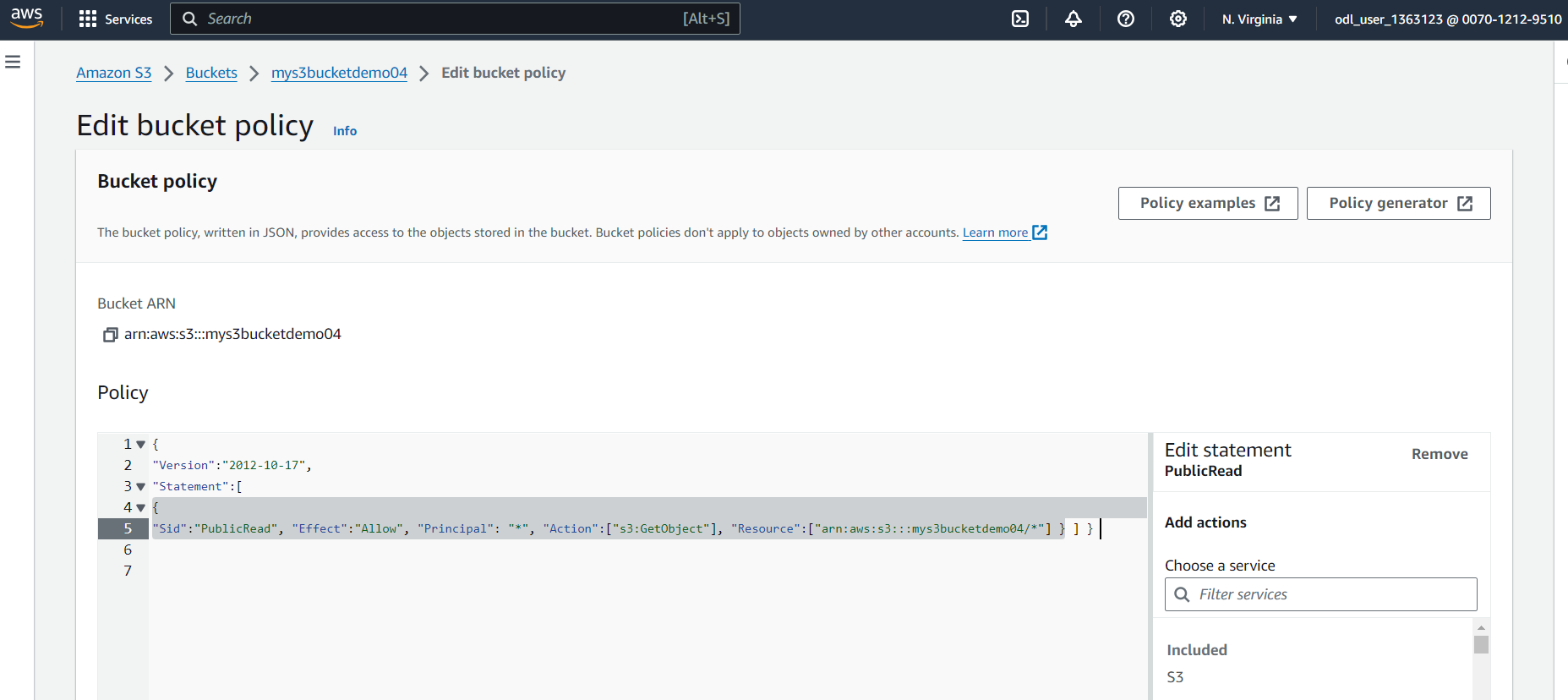
**{**

**"Version":"2012-10-17",**

**"Statement":[**

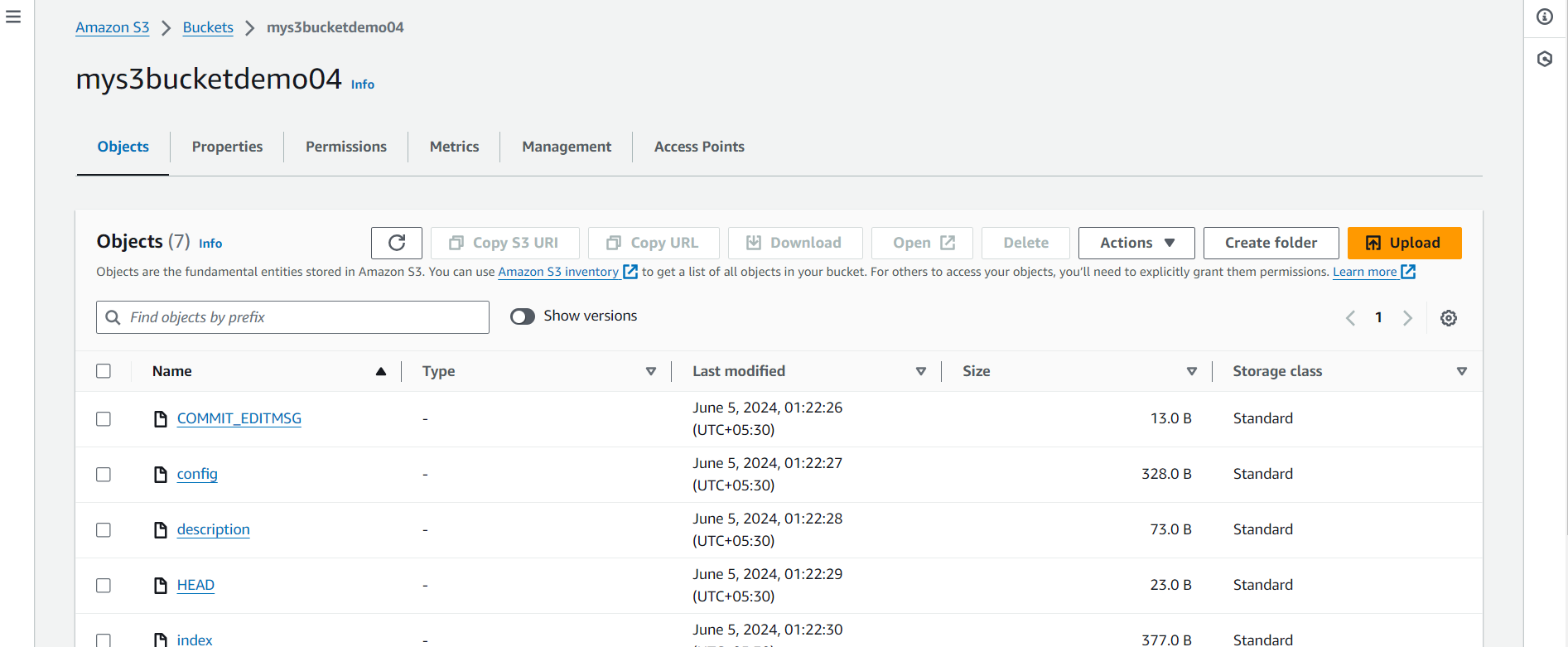
**{**

**"Sid":"PublicRead", "Effect":"Allow", "Principal": "\*", "Action":["s3:GetObject"], "Resource":["arn:aws:s3:::mys3bucketdemo04/\*"] } ] }**



**Step 4:** Hit the web app endpoint to check if the application is online

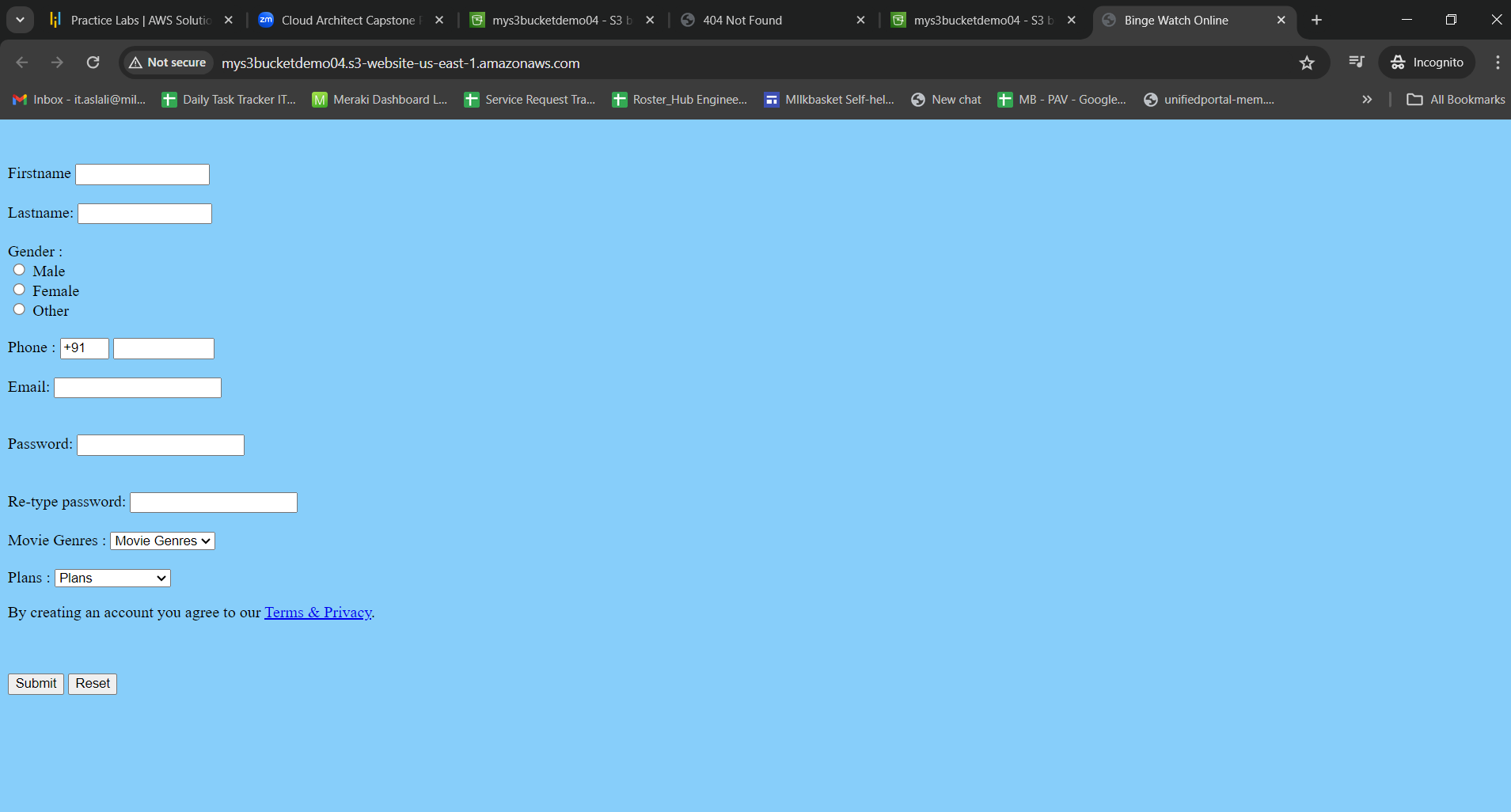
Step 4.1: Click on the Overview tab of the created bucket and click on the uploaded file



Step 4.2: Click on the **Object URL**

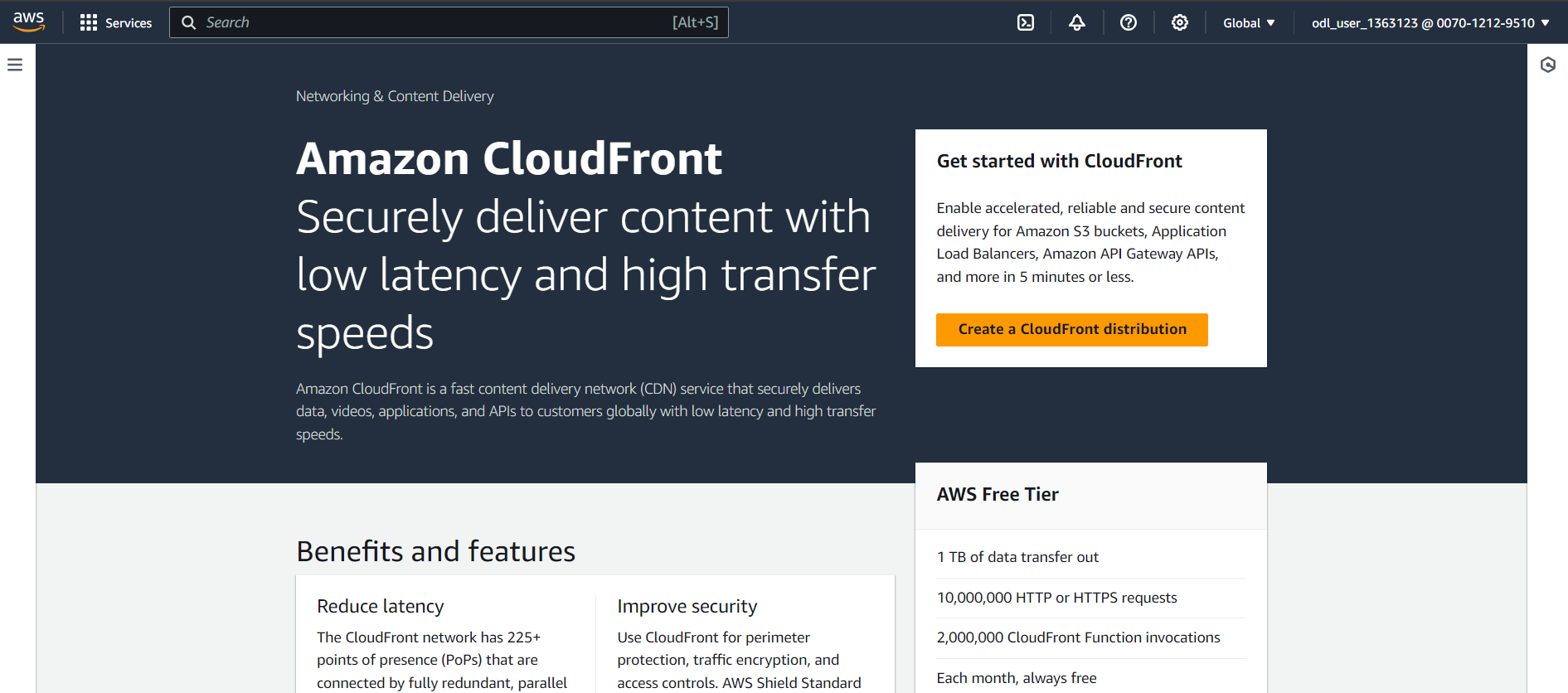
[http://mys3bucketdemo04.s3-website-us-east-1.amazonaws.com](http://mys3bucketdemo04.s3-website-us-east-1.amazonaws.com/)

Step 4.3: Your application will start running.

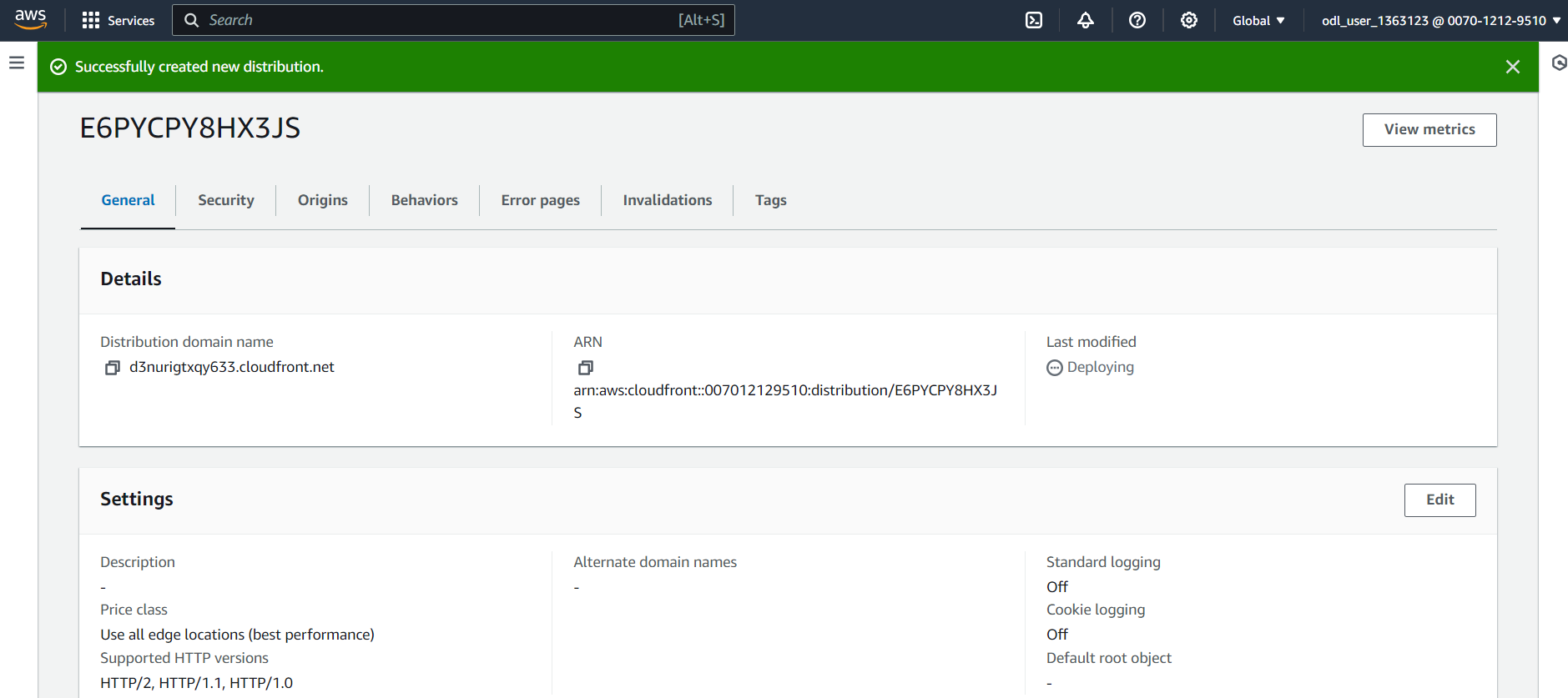


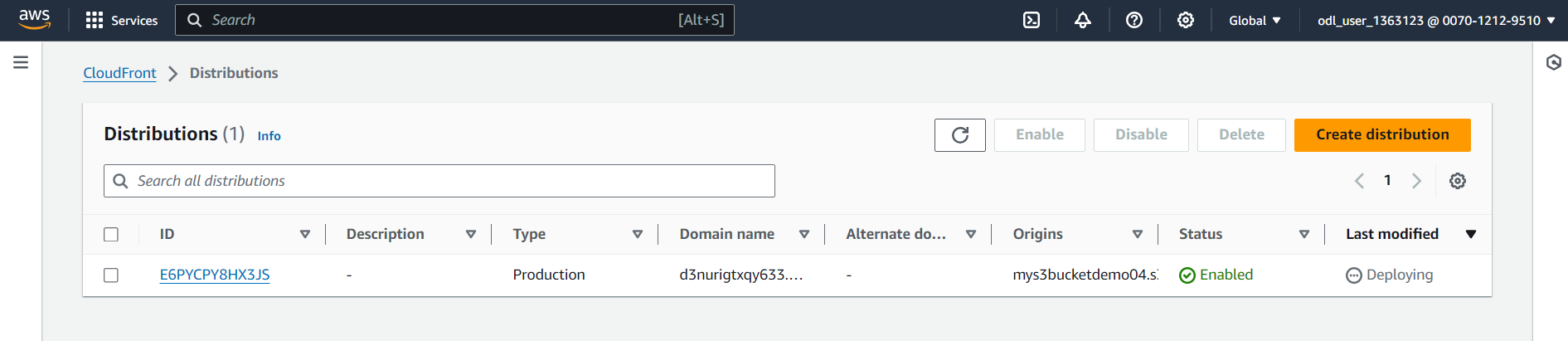
**Step 5:** Now create a CloudFront distribution corresponding to the static web app endpoint

Step 5.1: Go to services and search for CloudFront



Step 5.2: Click on **Create Distribution**





**Step 6:** Configure the CloudFront distribution to point to your domain by editing the configuration and adding the domain name in Alternate Domain Name field

Step 6.1: Once the CloudFront service get deployed, create a new file of .html extension in your system and copy the following content in it

<html>

<head>My CloudFront Test</head>

<body>

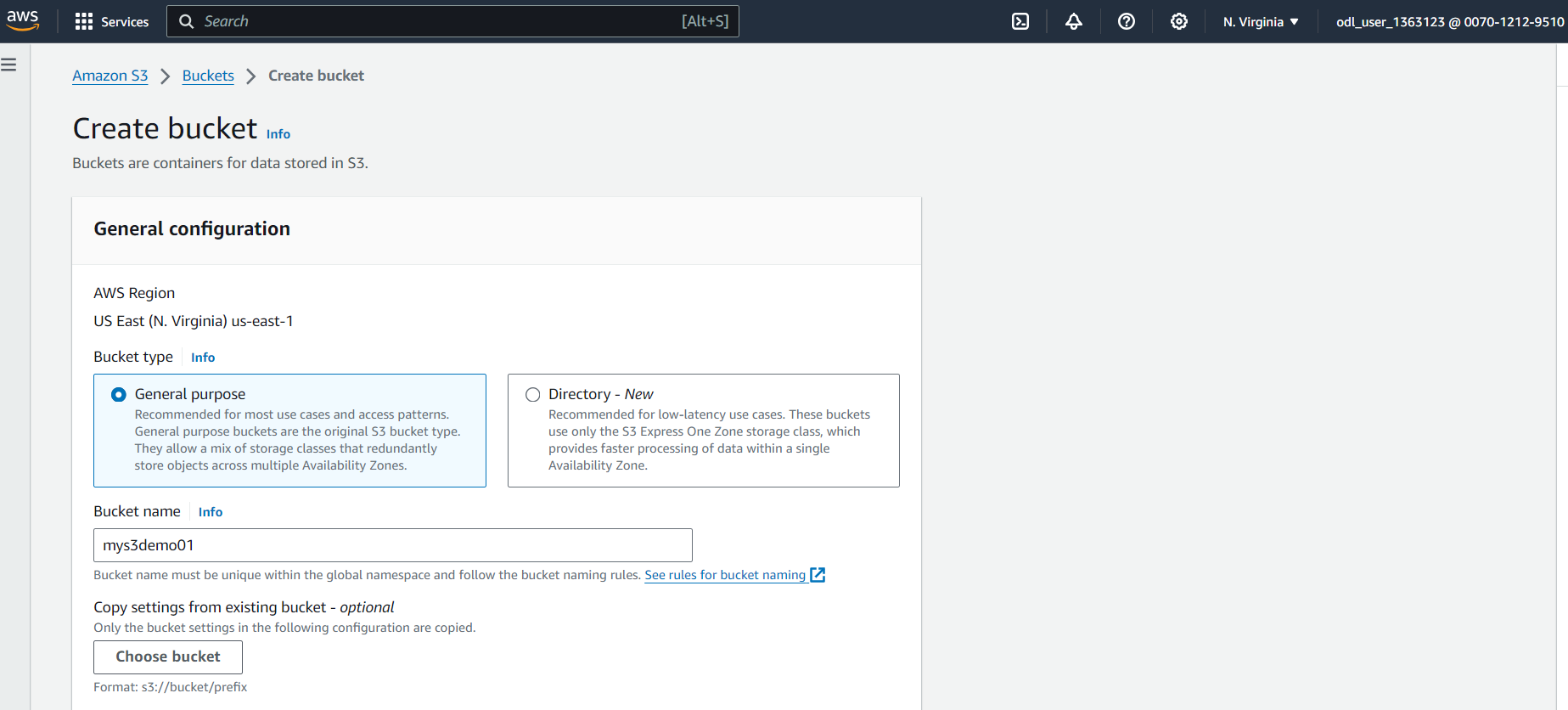
<p>My text content goes here.</p>

<p><img src=https://d111111abcdef8.cloudfront.net/.html alt="my test image"/>

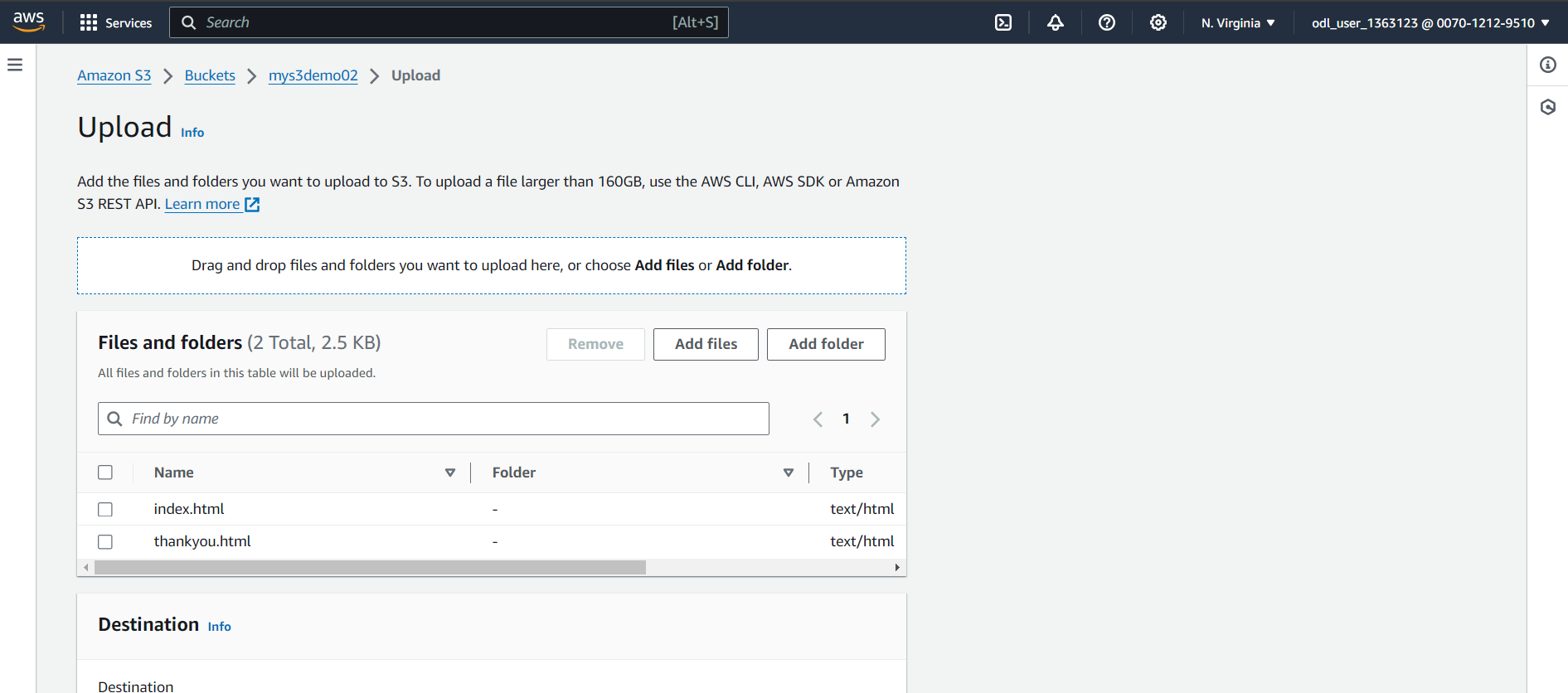
</body>

</html>

Step 6.2: Upload the same file in the created S3 bucket



Step 6.3: Click on the newly uploaded file, and go to the Object URL



Final Result is done.

