MASTERING AWS CLOUD

FROM BEGINNER TO PRO

Table Of Content

Introduction to Elastic IP

Introduction to AWS CLI

Introduction to Elastic Block Store (EBS)

How to Create an EBS Volume

Introduction to EBS Snapshots

Introduction to AWS ELB

How to Set Up AWS ELB

Introduction to AWS CloudWatch

Introduction to AWS EFS

Introduction to AWS Auto Scaling

How to Create an AWS Auto Scaling Group

Introduction to Amazon S3

How to Use AWS S3

Introduction to AWS RDS

Introduction to Data Center Structure

Imagine a data center with a number of computers and servers.

Hundreds and thousands of servers run side by side to provide compute resources to an organization and its branches.

Imagine all these servers are virtualized and a hypervisor is installed on each of them, so that the virtualization team can create virtual machines on them. If one of the employees in the company needs a virtual machine or a computer to do their work, they have to contact the virtualization team to get the required compute resources.

The Virtualization Team works in the data center, and most large companies require more than one administrator to manage their virtualization platform.

This is virtualization — something that existed before cloud computing.

Now you see similar virtualization platforms, but instead of contacting the virtualization team to get compute resources, you have a self-help portal, a website, or often a command-line interface to access the virtualized platform.

So if you need a virtual machine, virtual storage, or any other virtual service, you simply need to log in and create it yourself — this is called **cloud computing**, where access to your virtual resources is through the network. This means you can connect to your cloud portal via API from anywhere and at any time, and create, manage, or maintain your virtual resources.

Now, if this setup is for a single organization, it is known as a **private cloud**. But if it is publicly available so that anyone can sign up with a cloud provider to access it, it is known as a **public cloud provider**.

AWS, Azure, and Google Cloud are major names in the public cloud environment.

In this course, we use AWS Cloud Computing.

What is Cloud Computing?

It is the on-demand delivery of IT resources over the internet, where you pay only for what you use. Therefore, there is no need to purchase hardware.

You can access any computing power, storage, or database from a cloud provider like AWS.

In today's world, more than 90% of companies use cloud computing — and they do so because of the many benefits it offers.

Benefits of Cloud Computing

Some of the most important benefits of cloud computing include:

-Agility

You can easily access a wide range of cloud technologies and quickly build anything you can imagine.

You just need to create an account in AWS and start building your compute resources.

-Elasticity

You can easily scale your resources up or down based on your organization's needs.

You can access resources whenever you need and in whatever amount you require. However, since you grow or shrink your resources, you must keep control over your costs — something most people tend to forget.

-Cost Saving

Since you pay based on usage and consumption, it helps you save on costs.

-Deploy Globally in Minute

You can make your services global in less than a minute. For example, AWS infrastructure is spread across the world, and with just a few clicks, you can launch your application in multiple physical locations.

Placing your application closer to the end user can reduce latency and enhance the user experience.

Types of Cloud Computing

Cloud computing consists of the following three main types:

-Infrastructure as a Service

It is a virtual machine, and you can manage the virtual machine's operating system — this is what AWS's EC2 service provides.

-Platform as a Service

In this service, you don't have to worry about the virtual platform — you simply choose the platform you need. For example, if you need an Oracle Database, there's no need to create a virtual machine.

In this case, you can use the AWS RDS service, and AWS will provision and set up everything needed to run the Oracle Database — so you don't have to deal with the setup process yourself.

-Software as a Service

It is one of the cloud-based services that is easily accessible and usable — you just need to subscribe to it and start using it.

Introduction to AWS Free Tier

AWS is essentially an infrastructure that is spread across the globe, and you can build your own infrastructure on top of it. To do this, you need an AWS Free Tier account.

AWS is one of the largest cloud providers in the world today, holding nearly 46% of the market share.

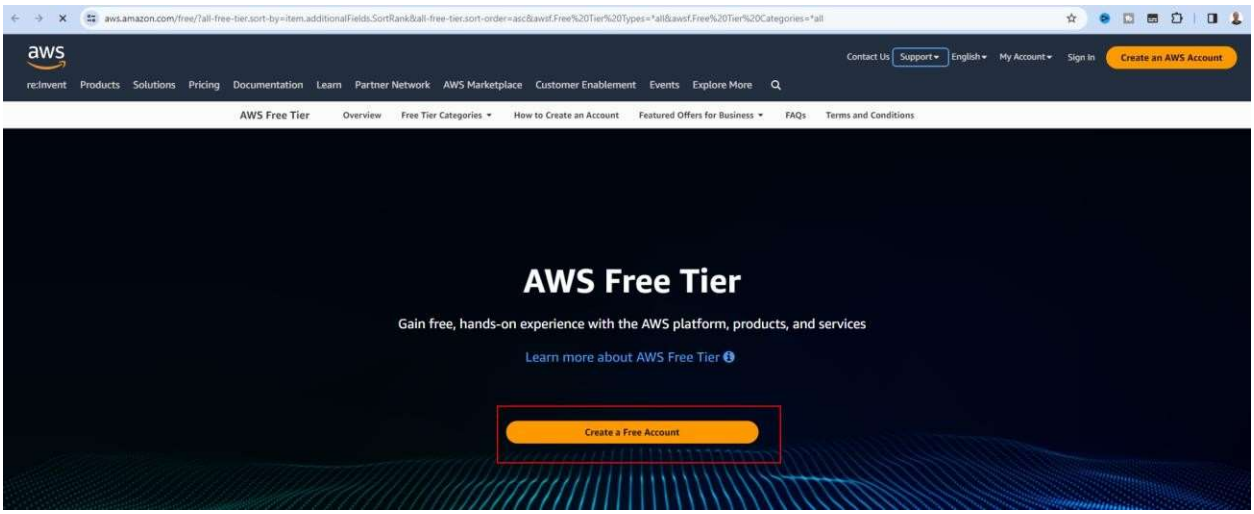
In this section, we intend to create an AWS Free Tier account, then modify its settings. Next, we will create an IAM user for accessing AWS services, enable MFA (Multi-Factor Authentication) on our AWS accounts, and configure a billing alarm. This alarm will notify us if usage exceeds the allowed limit — meaning if you create a service and keep it running without deleting it, charges will apply. However, by setting up a billing alarm, you'll receive a notification once your usage reaches the defined threshold.

Finally, you can configure a certificate for your domain within AWS using the **ACM (AWS Certificate Manager)** service. This allows you to use **HTTPS** for secure access to your web services.

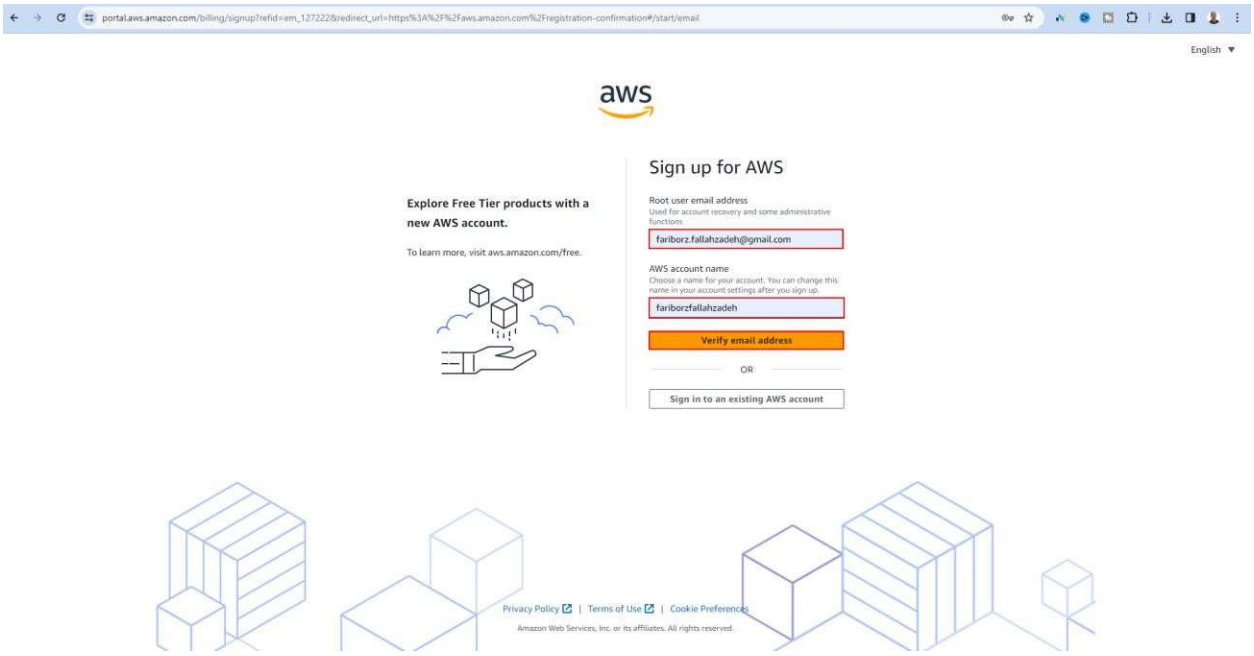
How to Create an AWS Free Tier Account

Search for **“AWS Free Tier”** on Google and then enter the Amazon website from the search results.

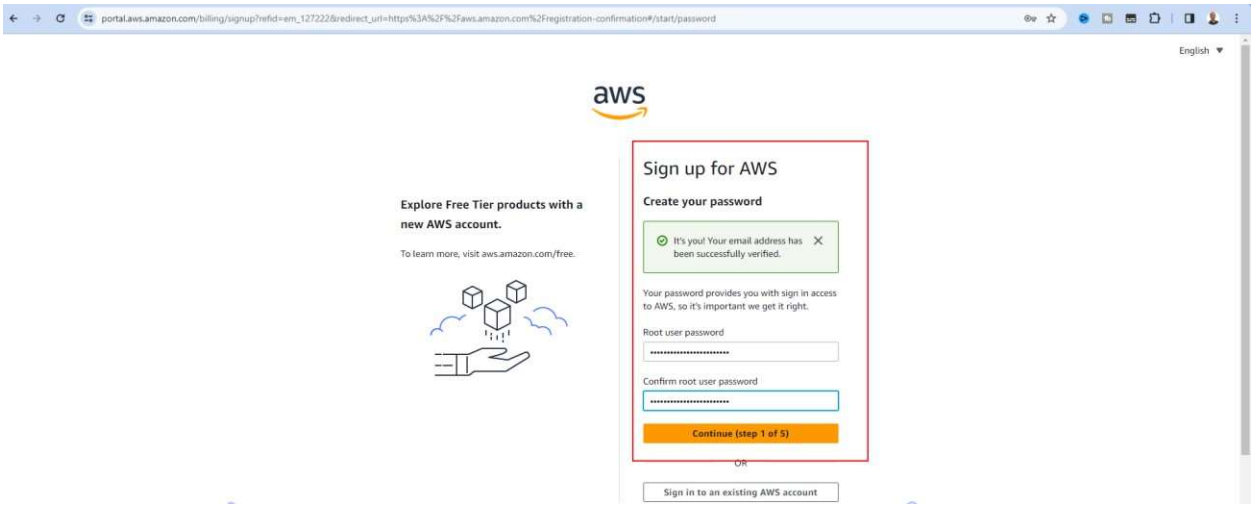
At this stage, click on the **Create Free Account** button.



At this stage, you need to provide a **Root User Email Address** and an **Account Name** to access the AWS Management Console.

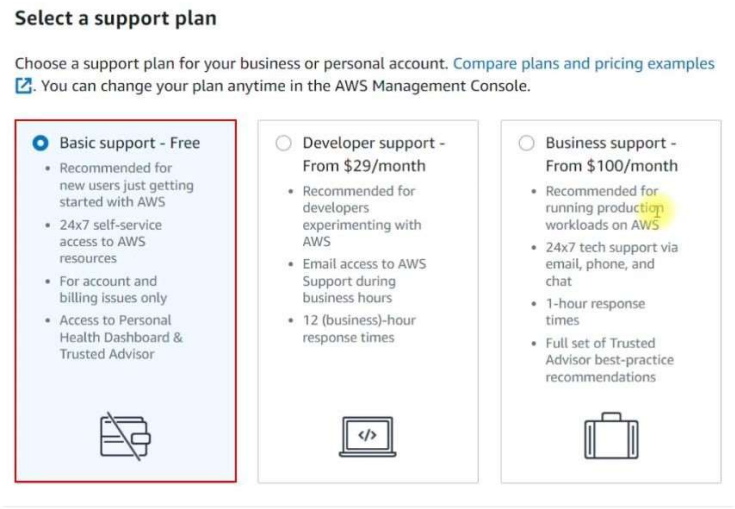


At this stage, your email address needs to be verified. An **Email Verification Code** will be sent to your email, which you must enter in the verification field, then click the **Verify** button. After that, you will need to set a **Root Password**.



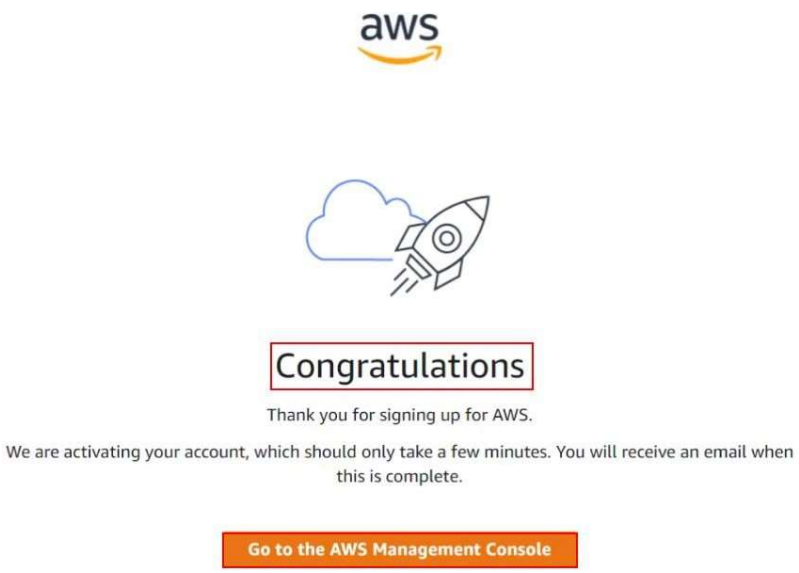
Next, you need to enter your **billing information** and **credit or debit card details**.

After completing your bank account and card information, you need to select a **Support Plan**. Since you are going to use this account for testing and learning purposes, it's best to choose the **Basic Support** plan, which is free.



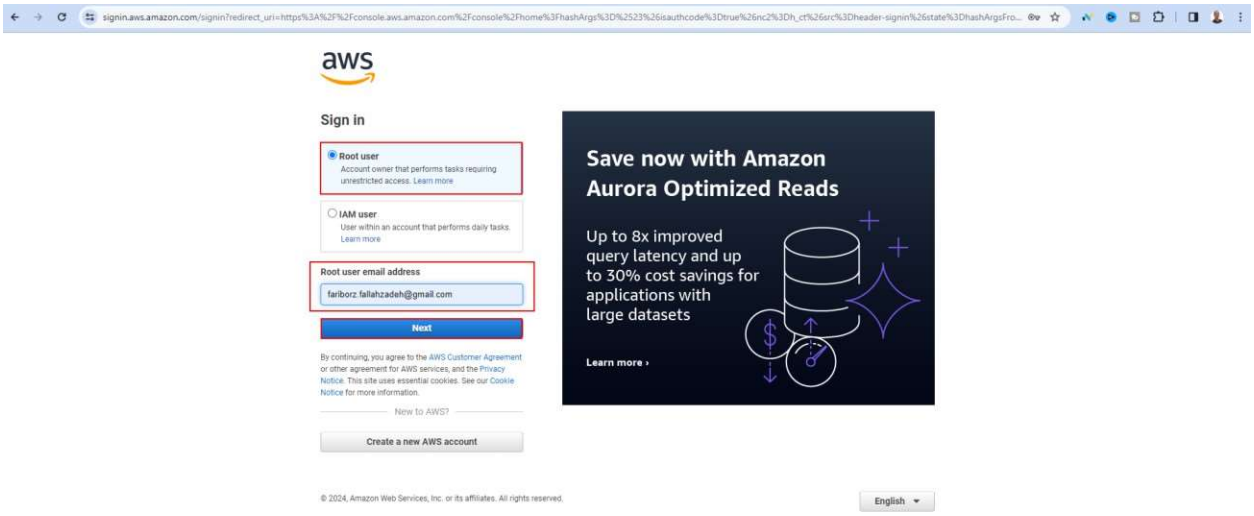
Finally, click on the **Complete Sign Up** button to finish the account creation process.

After completing the account creation steps, you should see the following message, which means your account has been successfully created.



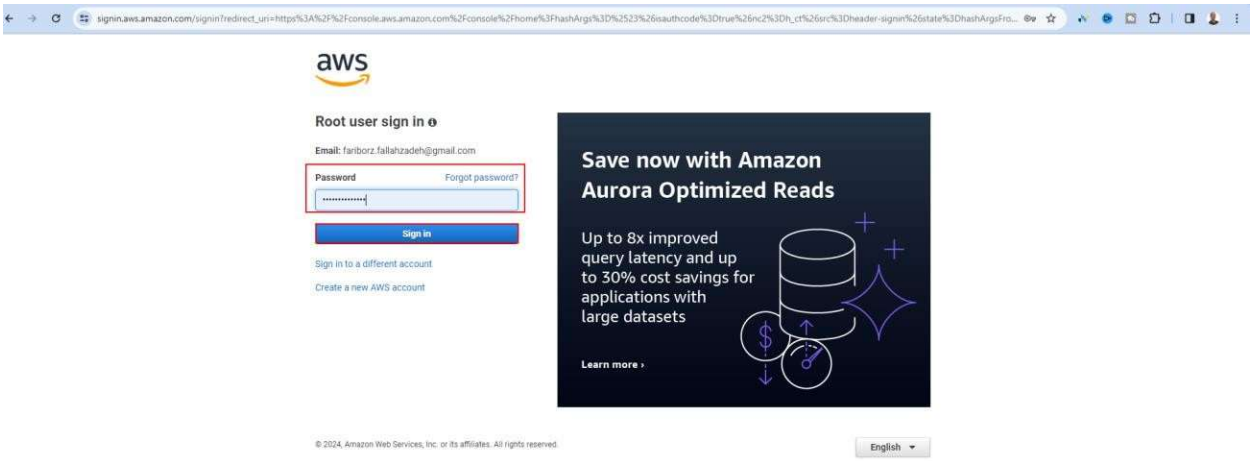
Then, click the button at the top to enter the **AWS Management Console**.

At this stage, you need to log in to your **AWS Management Console** using the **Root User**. The root user account has the highest level of access to the AWS Management Console.

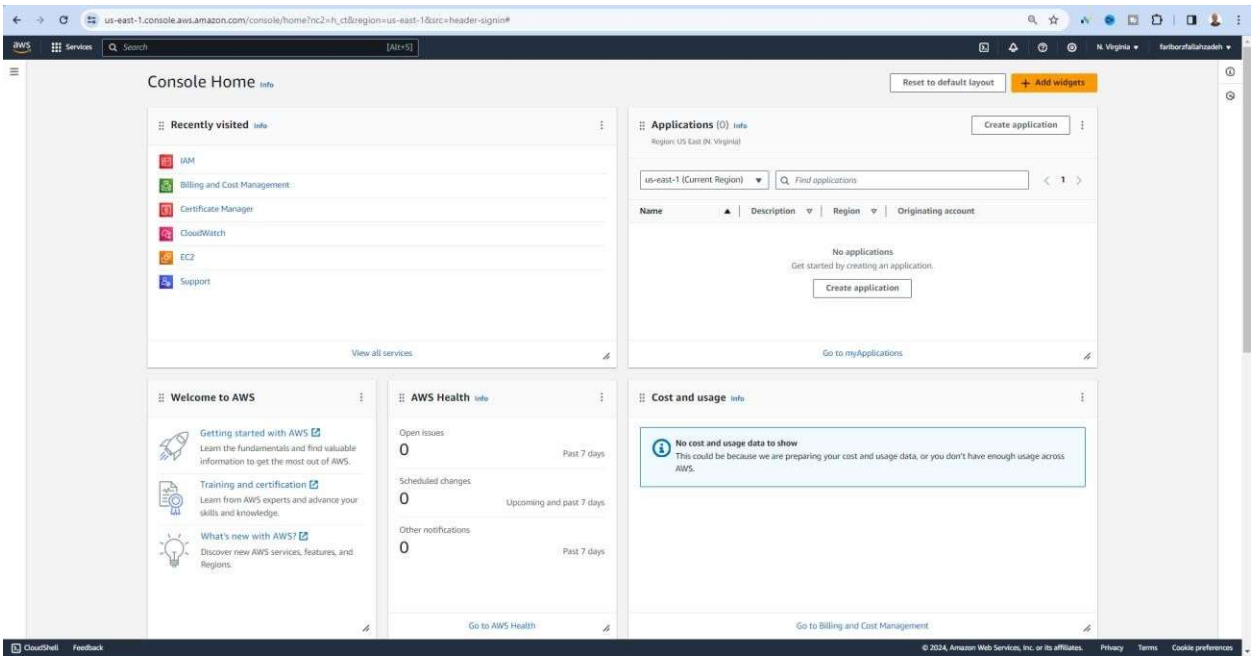


After entering the **Root Username**, click the **Next** button.

At this stage, you need to enter your **Root Password** and then click the **Sign in** button to access the AWS Management Console.

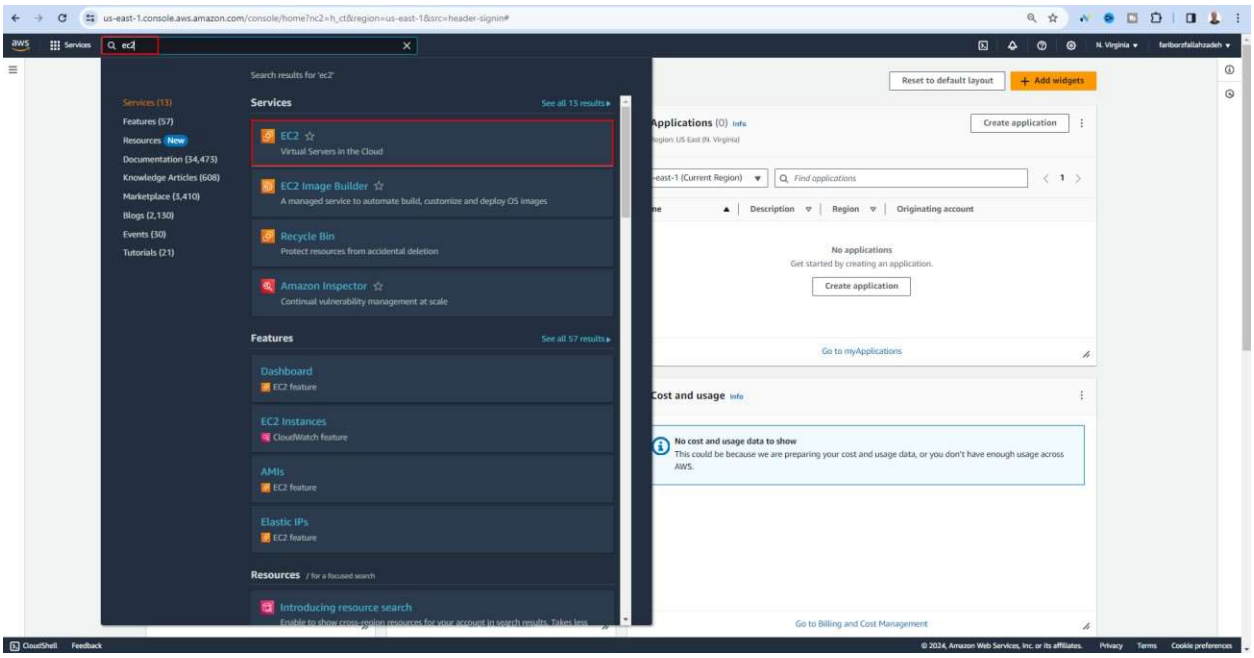


After logging in, as you can see, you are now inside the **AWS Management Console**.

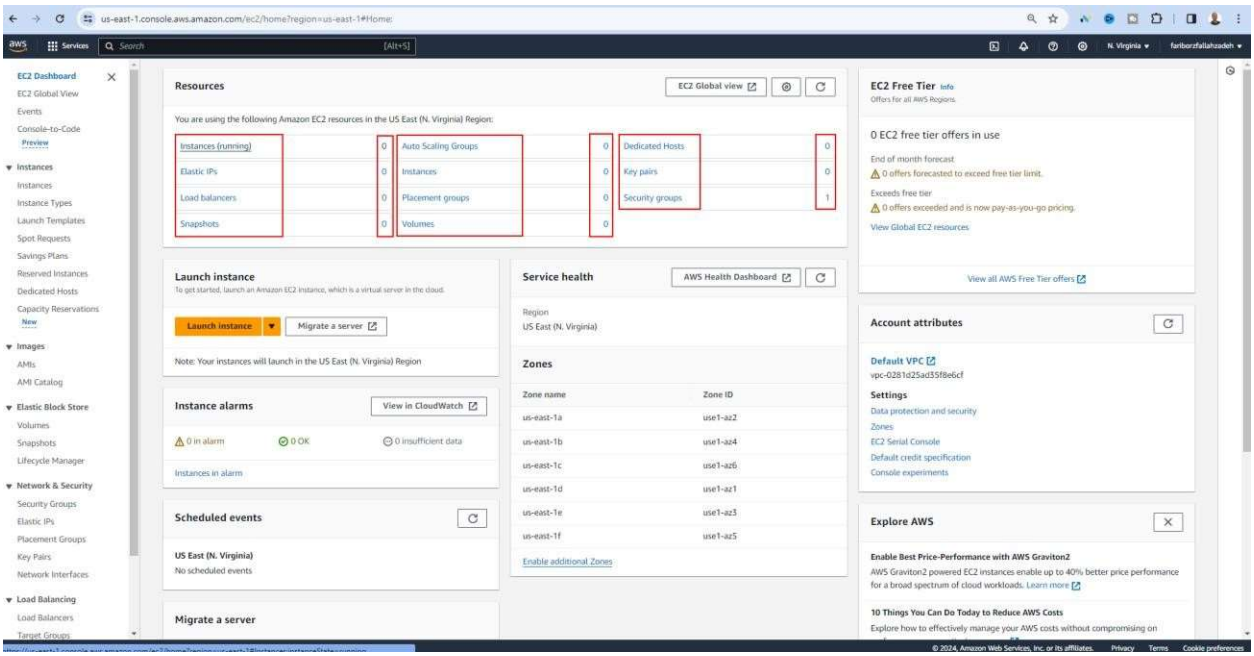


To make sure your account is activated, simply check one of the available services in your account — for example, the **EC2** service, which is used to create virtual machines.

At this stage, type **EC2** in the search bar and then click on it to enter the **EC2 Instances** page.

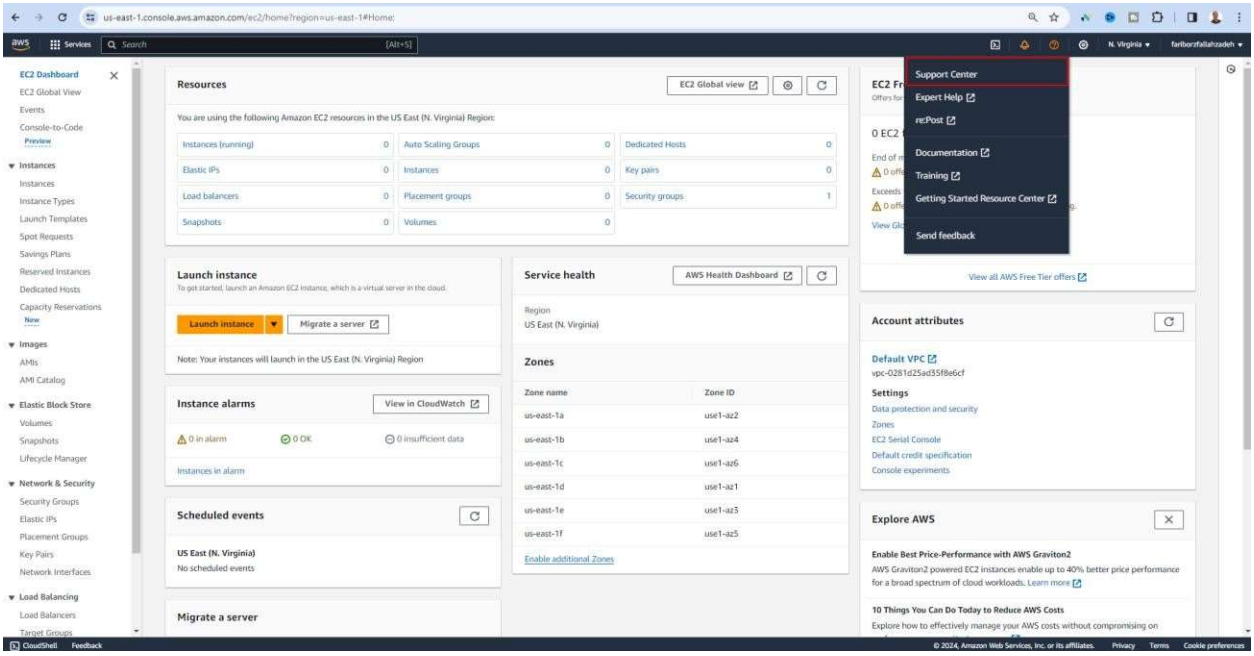


If all the values on this page are zero, it means your account is **active** and **enabled**.



If you see an error at the top or your account is not activated, you can submit a **support ticket** to AWS Support and explain the issue so it can be resolved as quickly as possible.

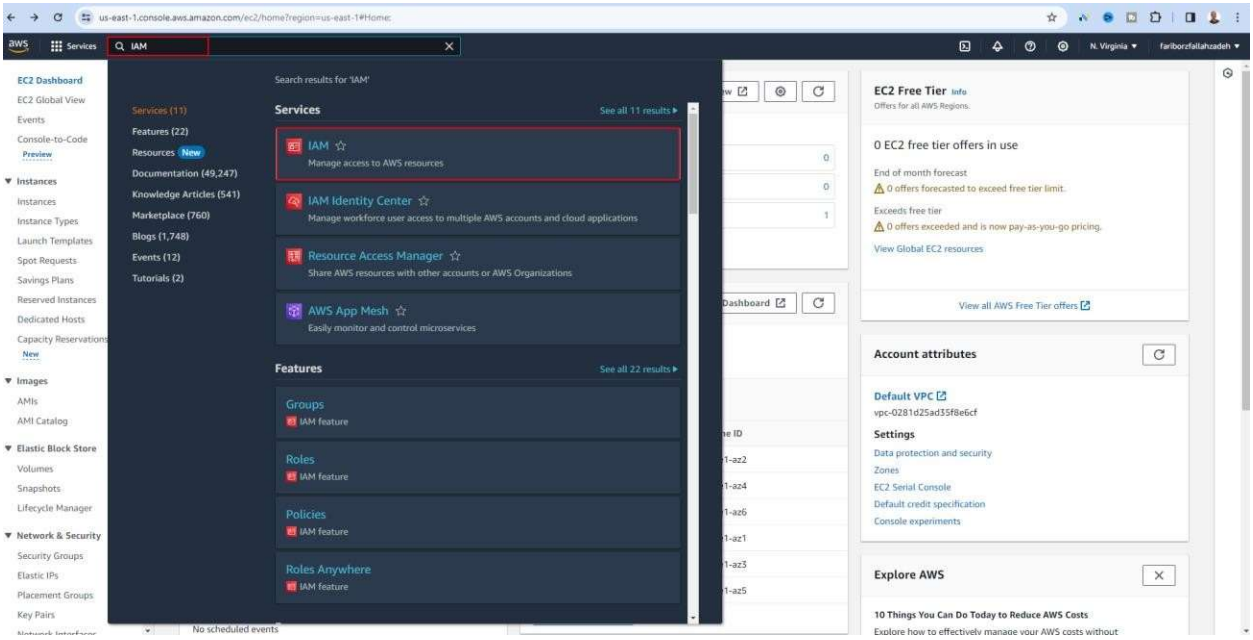
In the section below, you can access the **AWS Support Center** on the website.



It is recommended not to use the **Root User** to manage AWS services. Instead, it's better to use another user called an **AWS IAM User** for managing services.

How to Create an IAM User in AWS

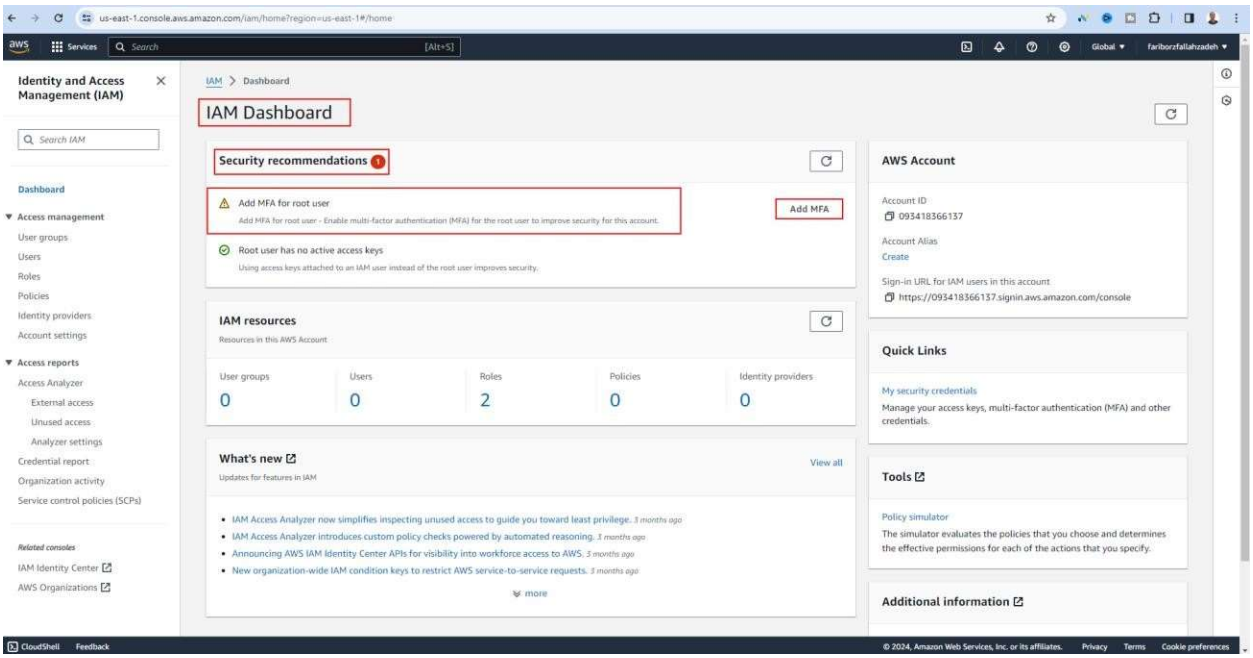
To create an IAM user, type **IAM** in the search bar and then click on it.



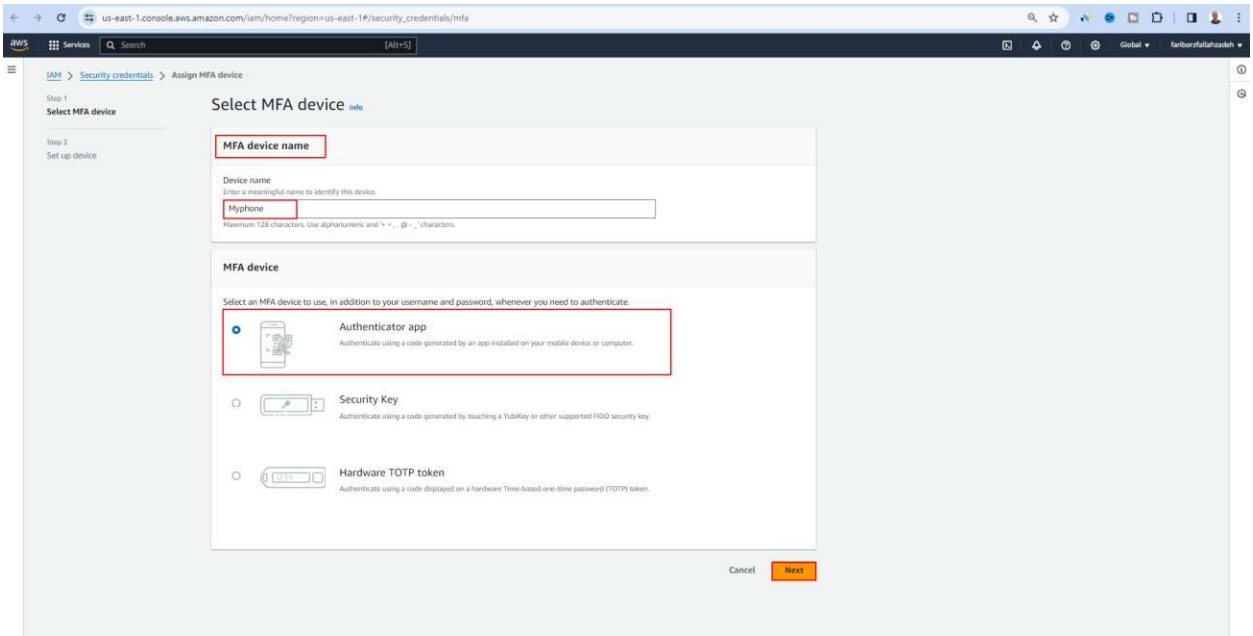
The first thing you need to do before creating an IAM user is to **enable MFA (Multi-Factor Authentication)** for the Root User. You should install an app called **Google Authenticator** on your phone. This app provides a 6-digit code used for authentication.

Each time you enter your password, you must also enter this code for authentication. This mechanism uses **OTP (One-Time Password)**, which means a unique numeric password is generated for you at each login attempt.

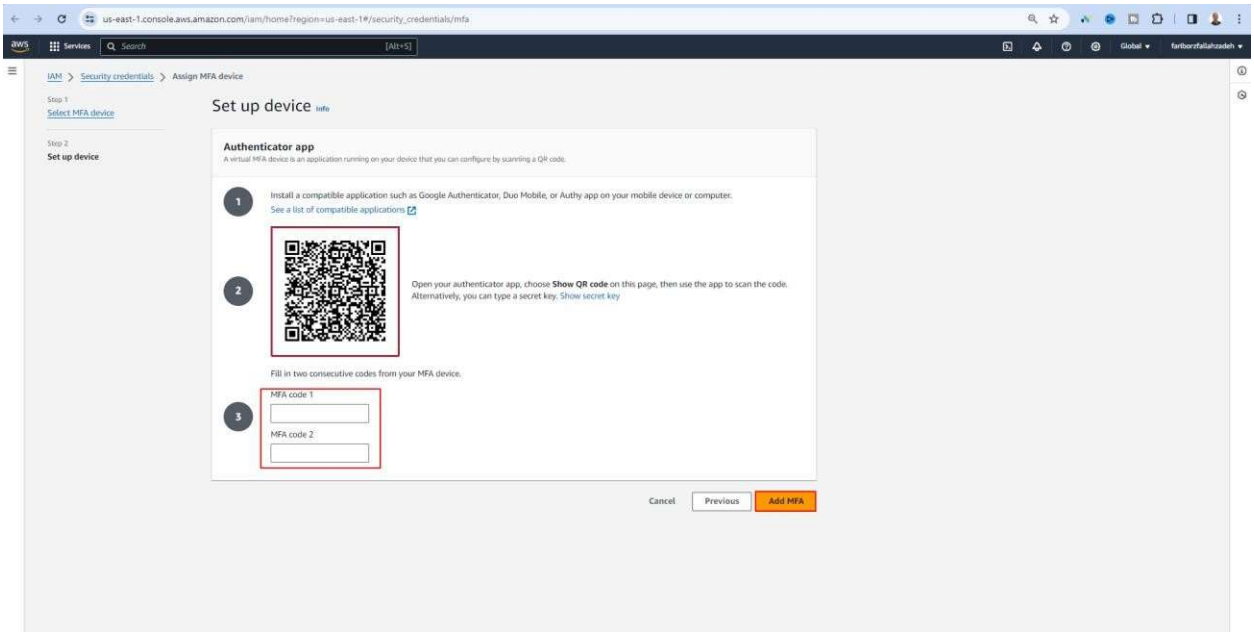
First, click on the **Add MFA** button.



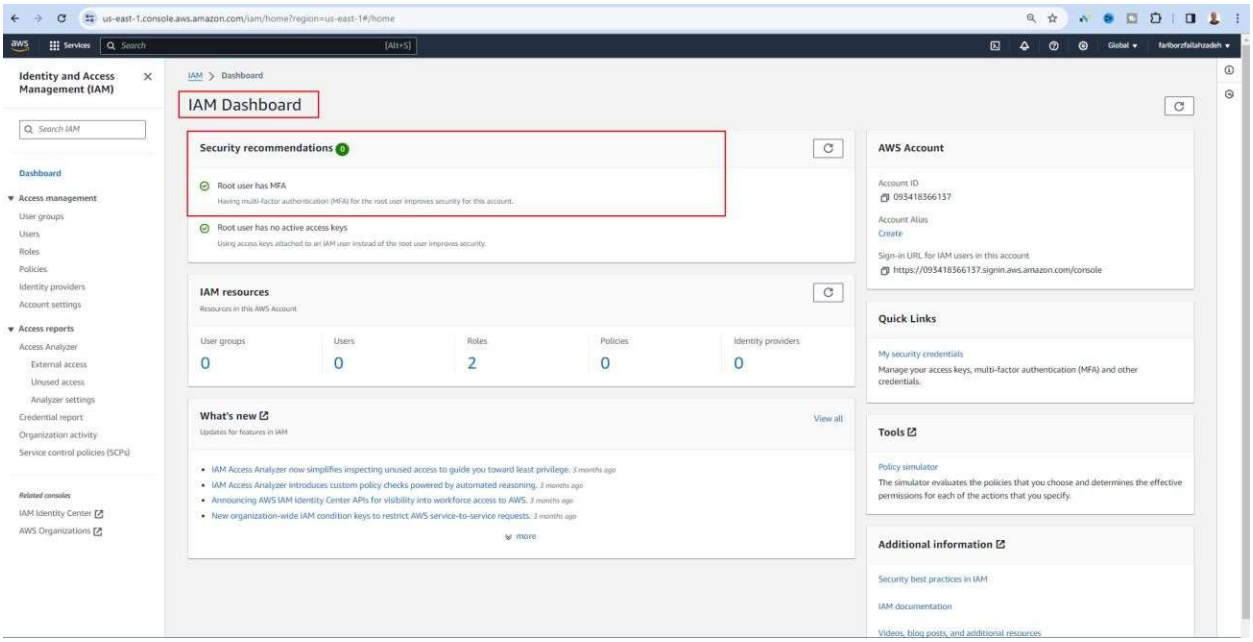
In the **Assign MFA Device** section, click to proceed and select an MFA device. Here, you should use your **phone** and assign it a name. Choose the **Authenticator app** option, then click the **Next** button.



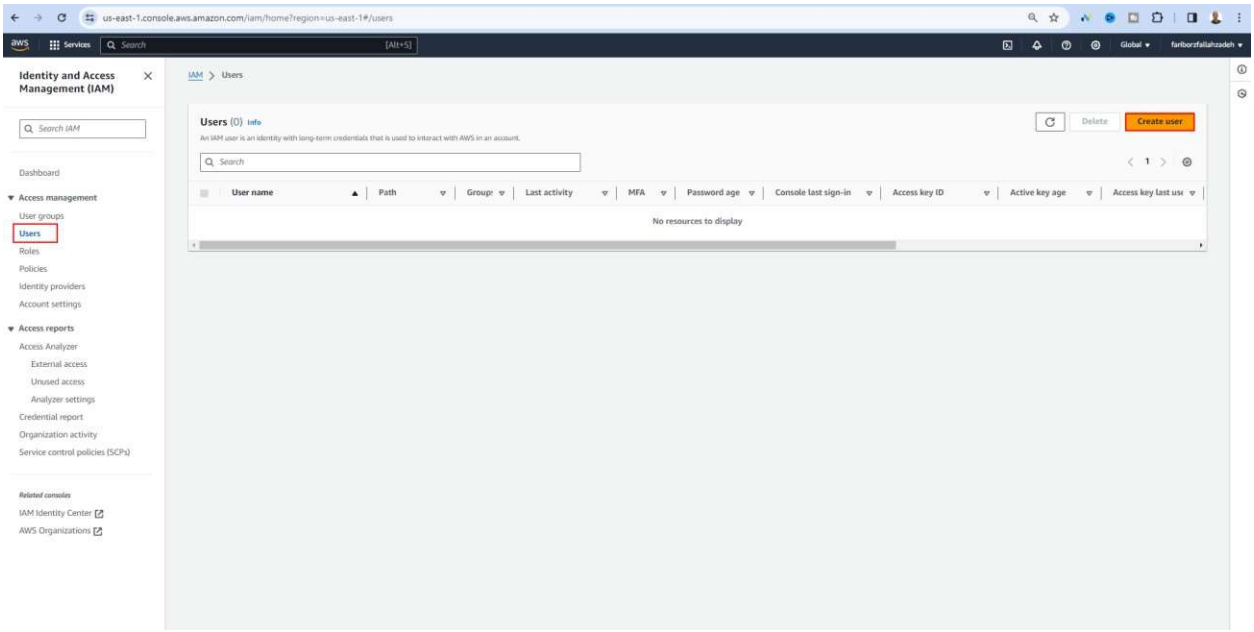
At this stage, open the **Google Authenticator** app on your mobile device, tap the + button, and in AWS click on **Show QR Code**. Scan the QR code with Google Authenticator, then enter the **two MFA codes** generated by the app into the fields provided. Finally, click on the **Add MFA** button.



Then, in the **IAM User** section, click on the **Dashboard** link and press the **Refresh** button. As you can see, **MFA has now been added** to the Root User.



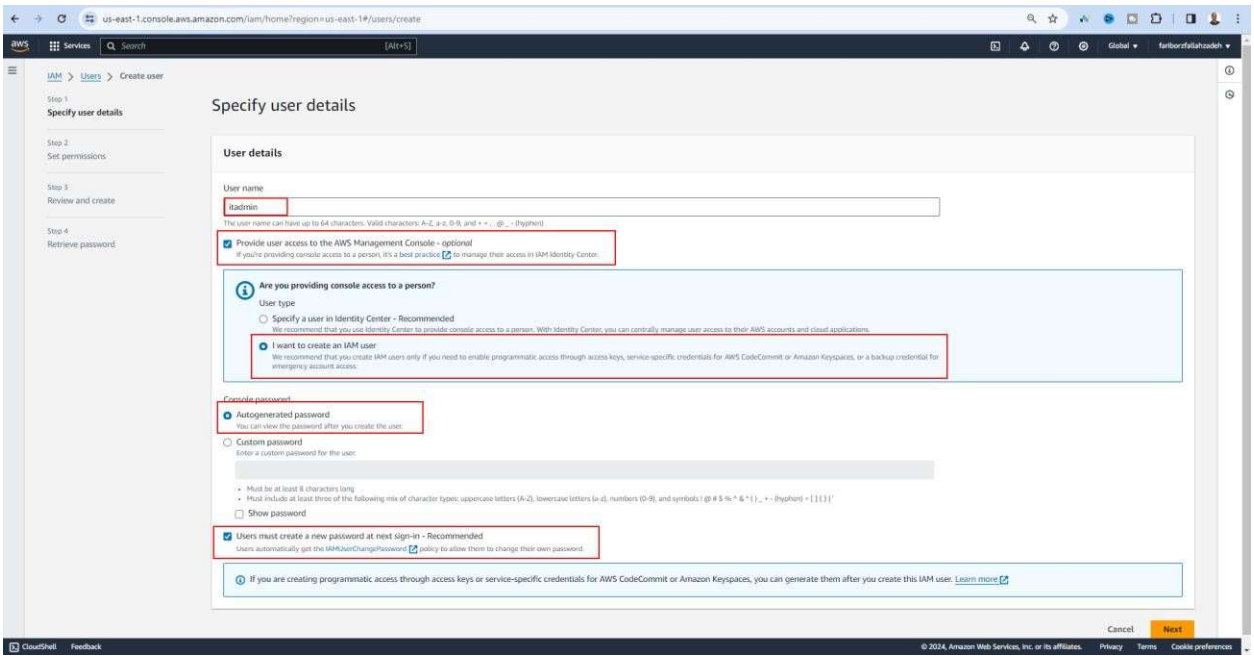
At this stage, to create an IAM user, click on the **Users** link and then click the **Add Users** button.



At this stage, choose a **username** for the IAM user, then select the option **"Provide user access to the AWS Management Console"** so the user can access the AWS Management Console, which is browser-based.

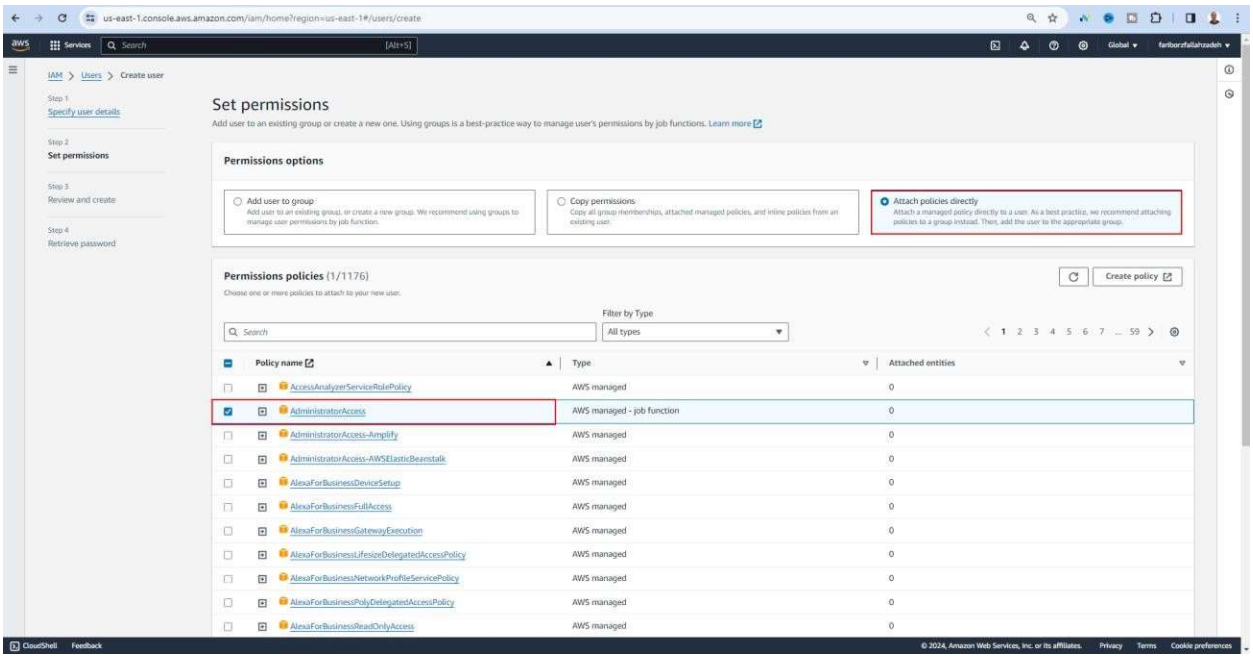
Next, select **"I want to create an IAM user"**, then choose **"Autogenerated password"**, and also check the box **"User must create a new password at next sign-in"**.

Finally, click the **Next** button.

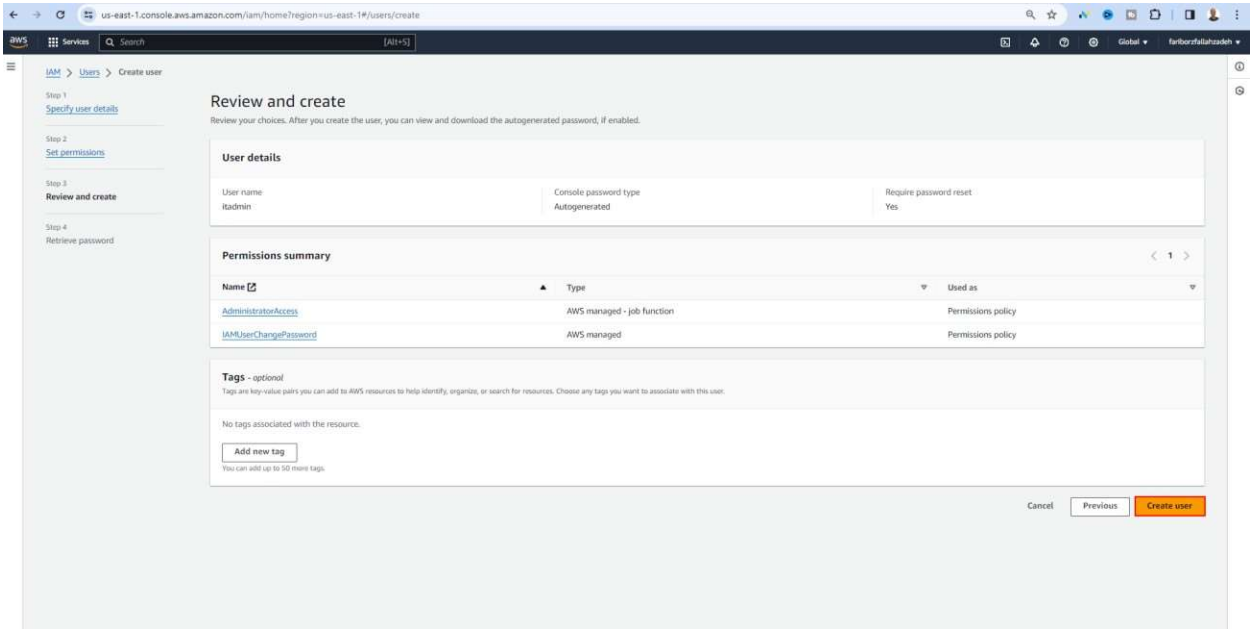


By default, this user has no permissions. To assign permissions, you can select the **Attach policies directly** option and then grant **AdministratorAccess** to the user.

As you can see in this section, you can apply various types of **policies** to the IAM user. Once selected, click the **Next** button.

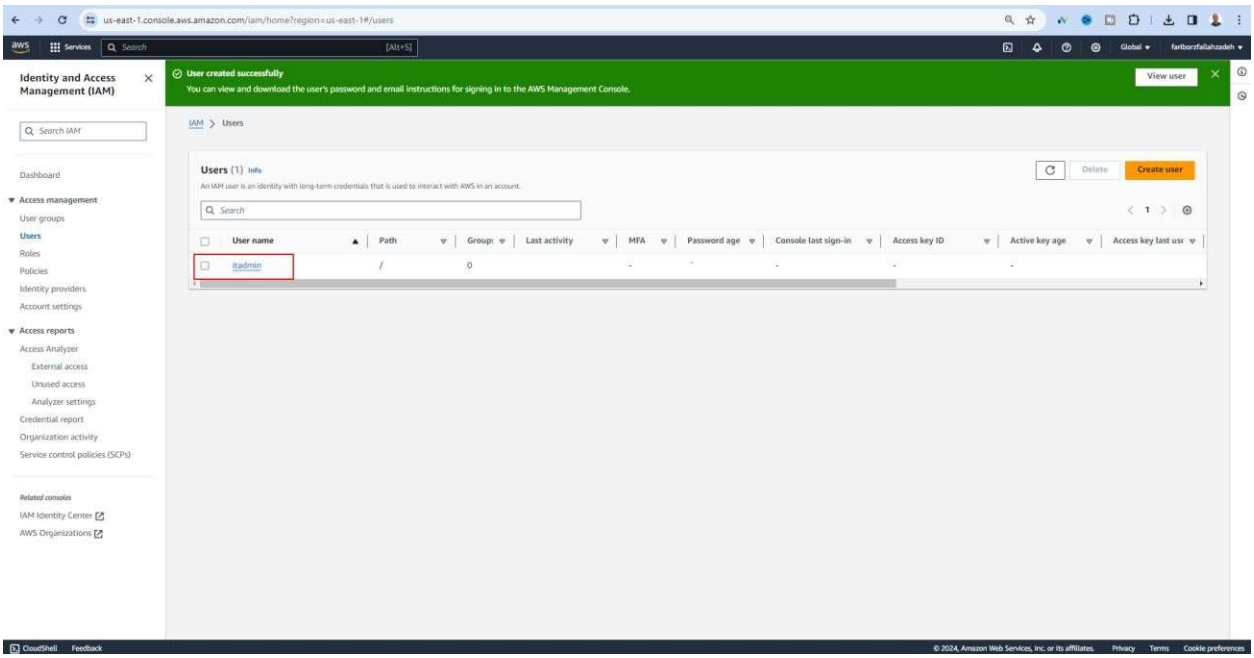


At this stage, click on the **Create User** button to create the IAM user.

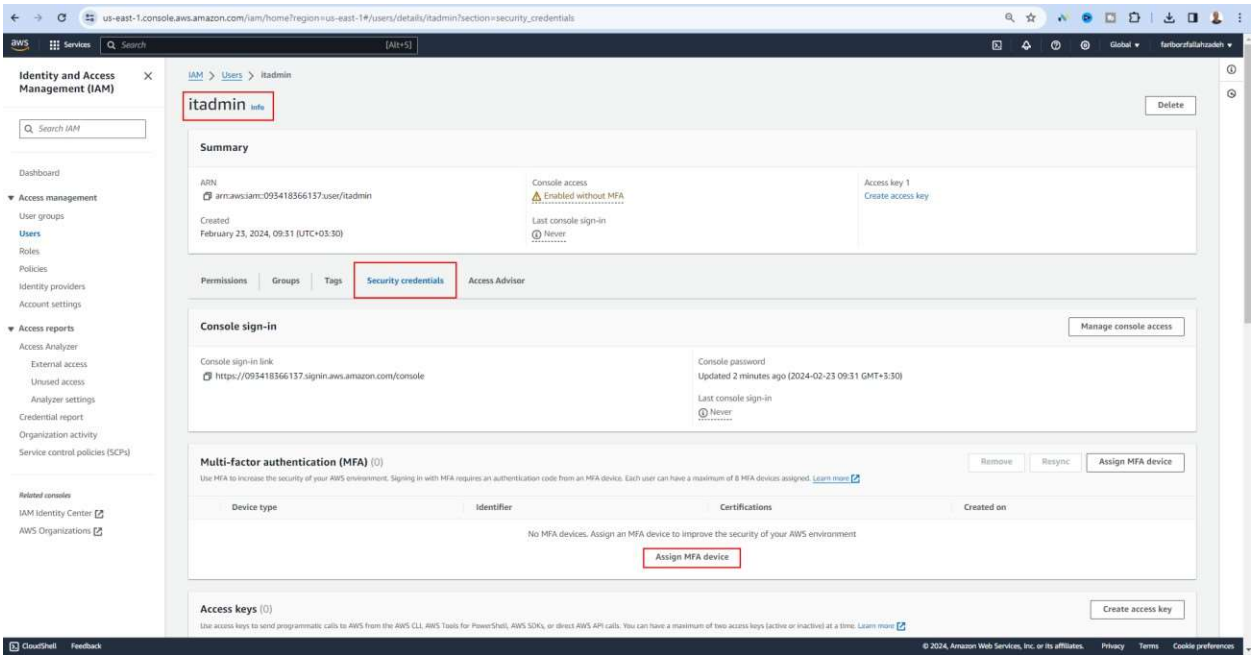


To view the IAM user, simply click on the **Users** link, and you will be able to see the IAM user you created.

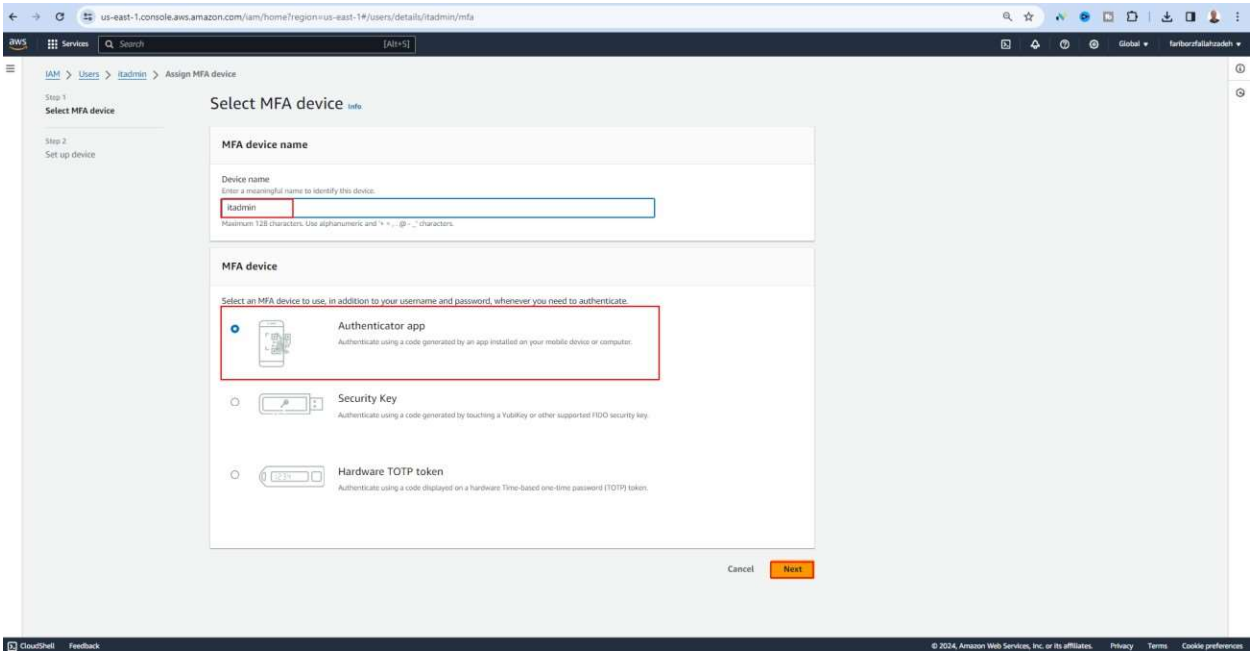
To enhance the security of the IAM user, it is recommended to add MFA to the user. To do this, simply click on the **User** link.



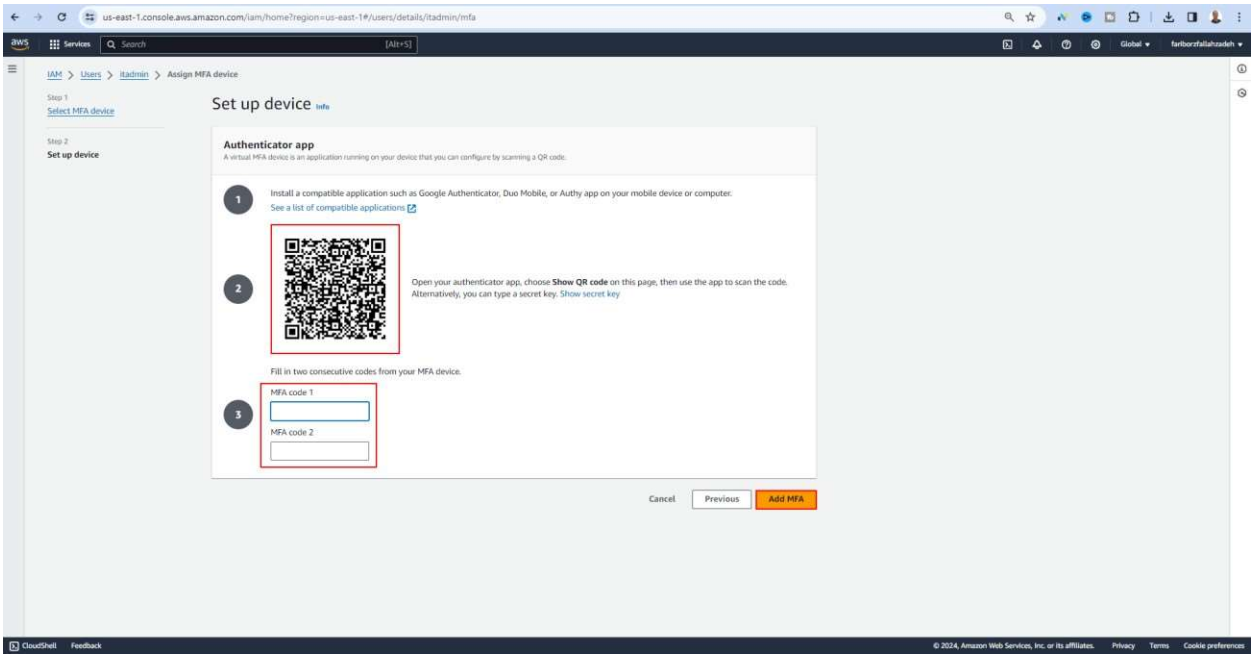
Next, click on the **Security Credentials** section and then click on the **Assign MFA Device** button.



At this stage, you need to specify a name for the **MFA Device**, select the **Authenticator App** as the MFA device type, and then click the **Next** button.



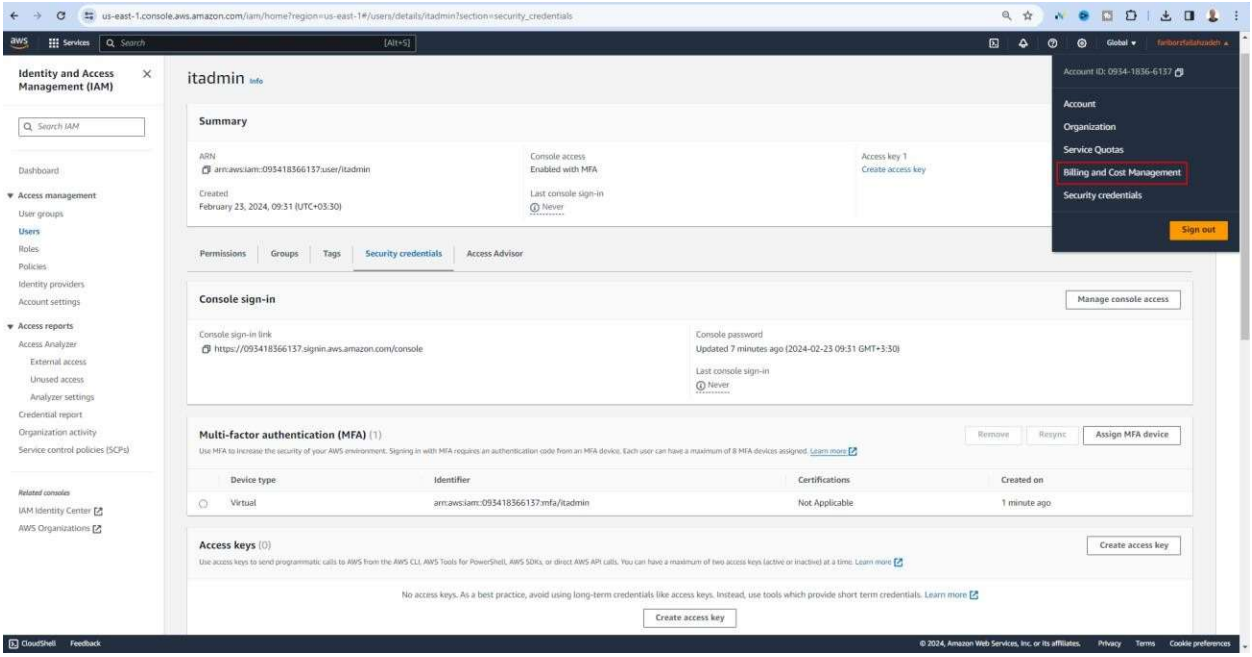
Then, go to your phone, open the **Google Authenticator** app, tap the + button, and scan the **QR code** provided by AWS. Enter the two **MFA codes** generated by Google Authenticator in the respective fields, and finally, click the **Add MFA** button.



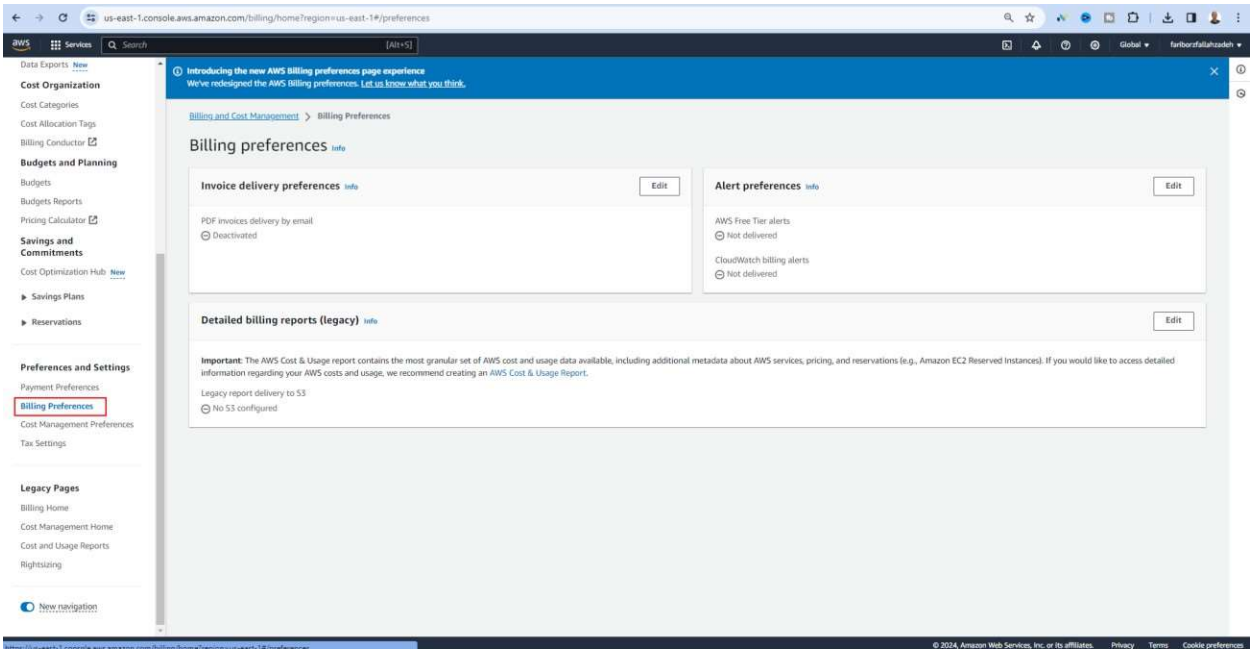
How to Set Up CloudWatch in AWS

To monitor the usage of services in AWS and control costs, you need to configure a few options under the **Billing and Cost Management** section. This will allow you to set up a **Billing Alarm**, which will notify you when usage exceeds the specified threshold. Additionally, you should delete any created resources to avoid unnecessary charges.

To configure the billing settings, you need to go to the **Billing and Cost Management** section.



At this stage, click on the **Billing Preferences** link.

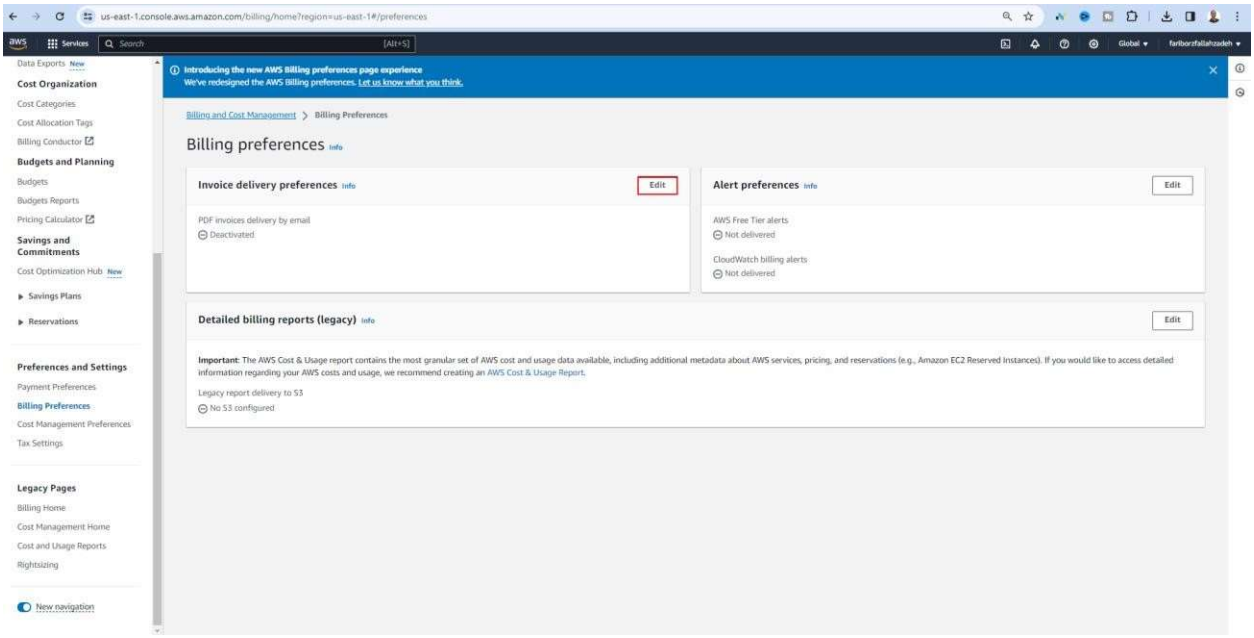


When using a Free Tier account, it's important to know that there are limits on the usage of resources. This means that if you exceed the allowed limits for services, you will be charged.

One of the things you should do is check the **Bill** section on the AWS website every night before going to bed. Review your bill to see the charges applied for using services, and if necessary, delete any resources each night to avoid incurring additional costs.

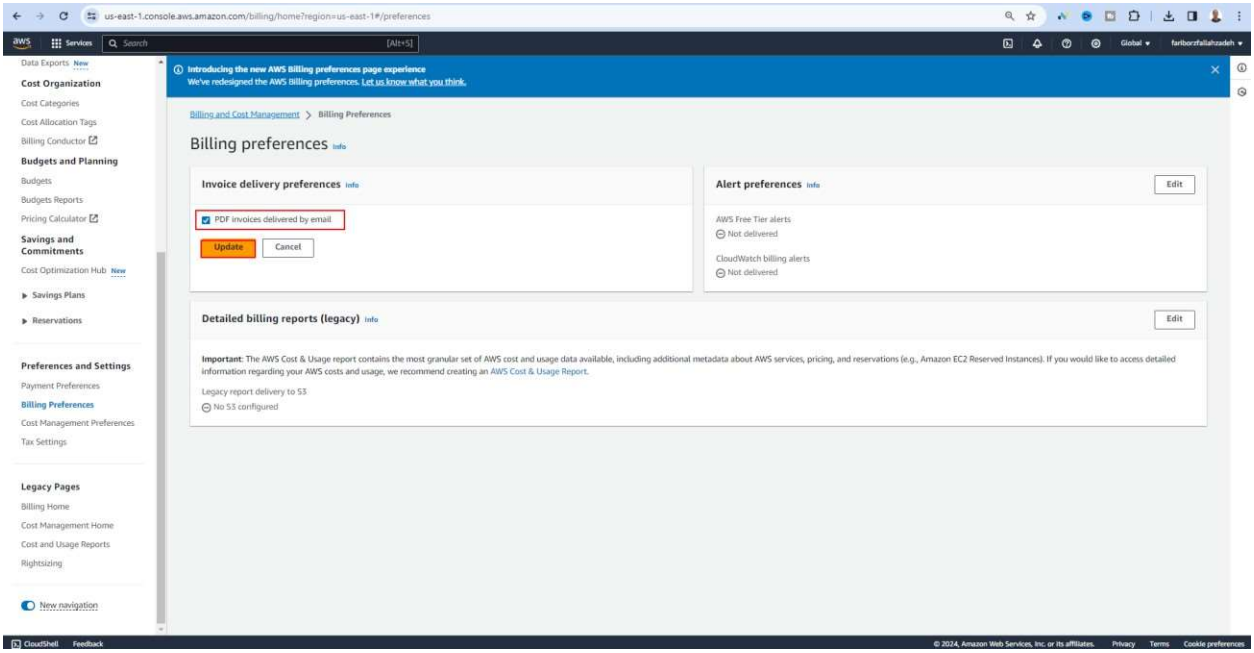
In the **Billing Preferences** section, it is recommended to configure the following options:

First, under **Invoice Delivery Preferences**, select the **PDF Invoices delivered by email** option, then click the **Update** button.

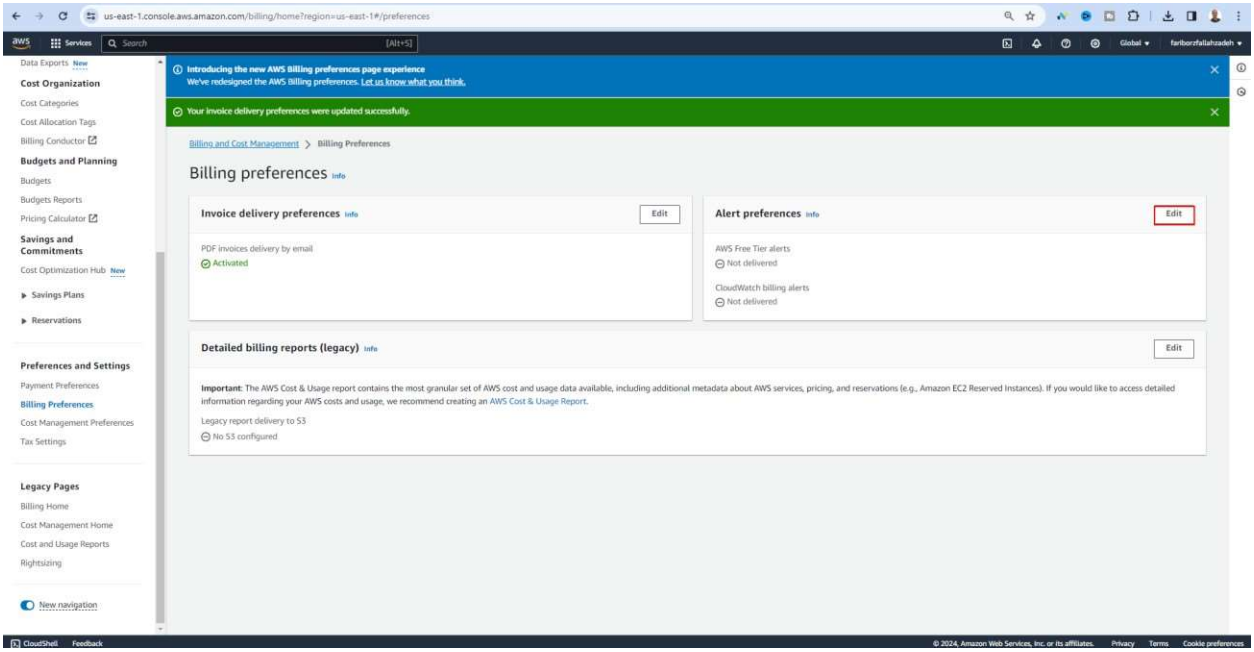


First, under **Invoice Delivery Preferences**, select the **PDF Invoices delivered by email** option, then click the **Update** button.

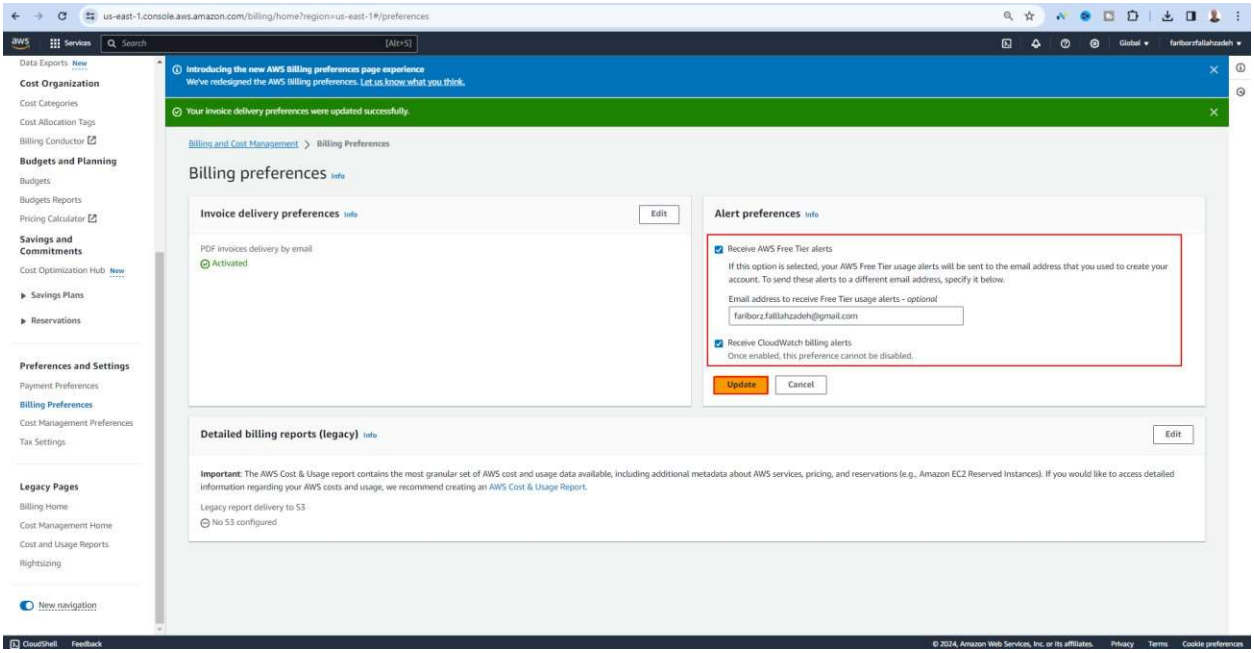
By selecting this option, your invoices will be sent to you via email as a PDF file.



Next, configure the settings for **Alert Preferences**.

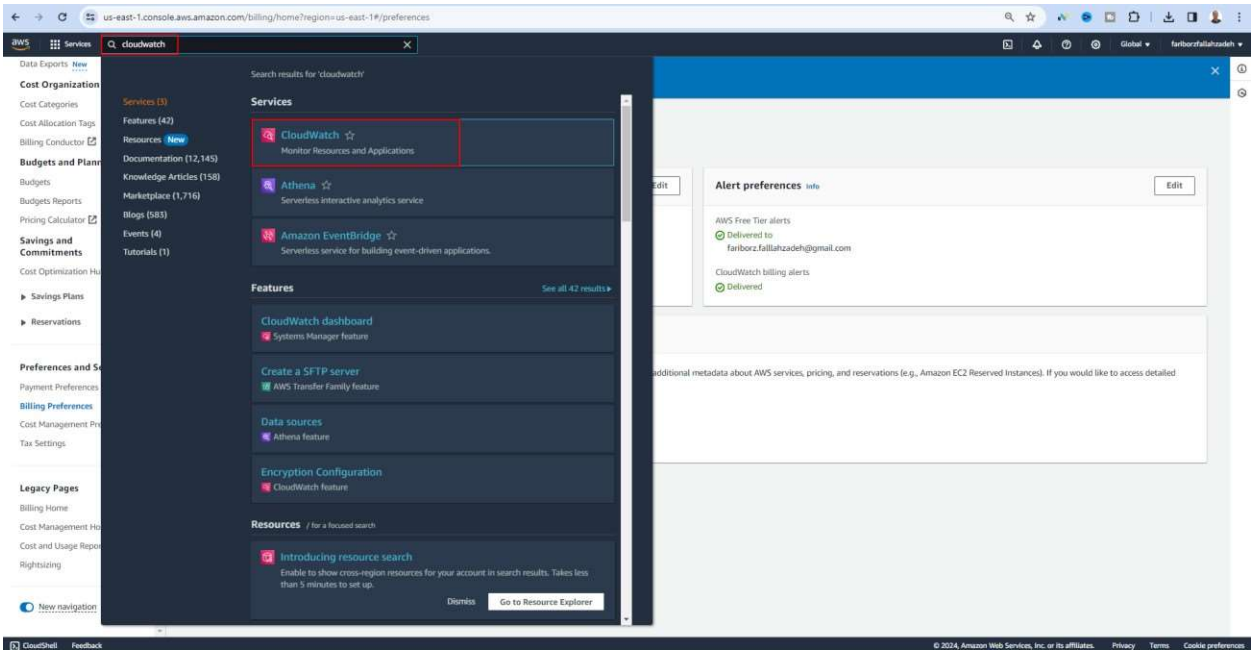


At this stage, by selecting the following options, you will receive an email via AWS **CloudWatch** when 85% of the services in your Free Tier account are used.

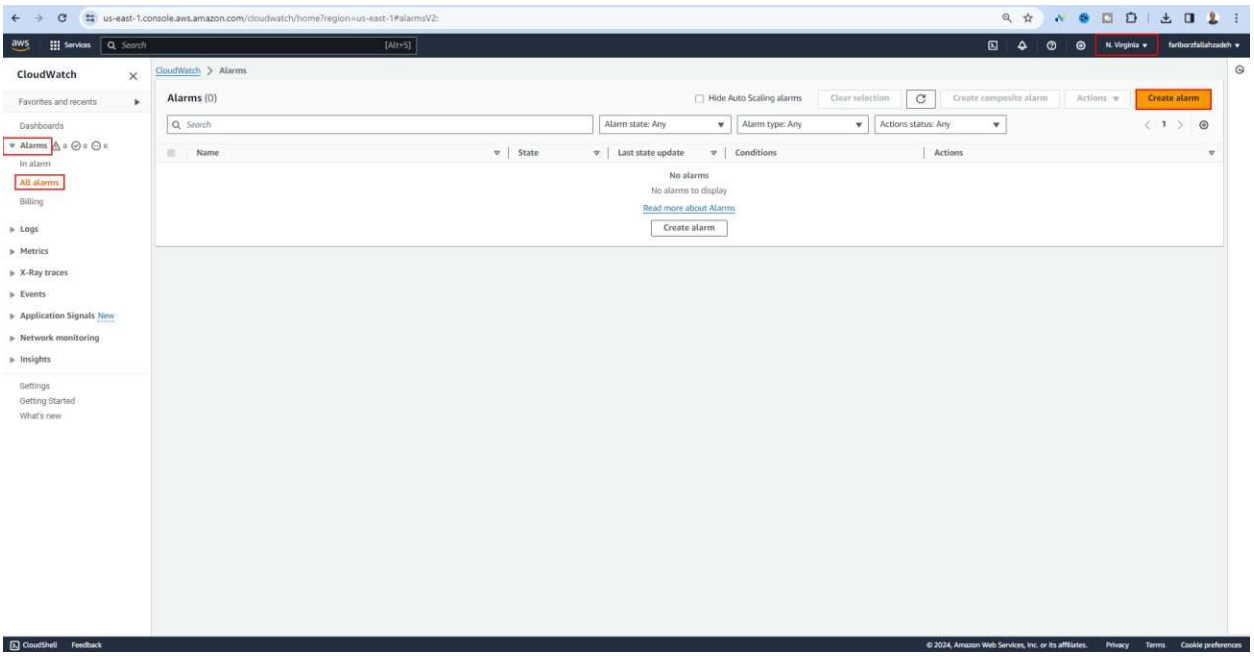


At this stage, you need to configure the settings for **CloudWatch**. CloudWatch is a monitoring service, and we need this service to monitor your billing.

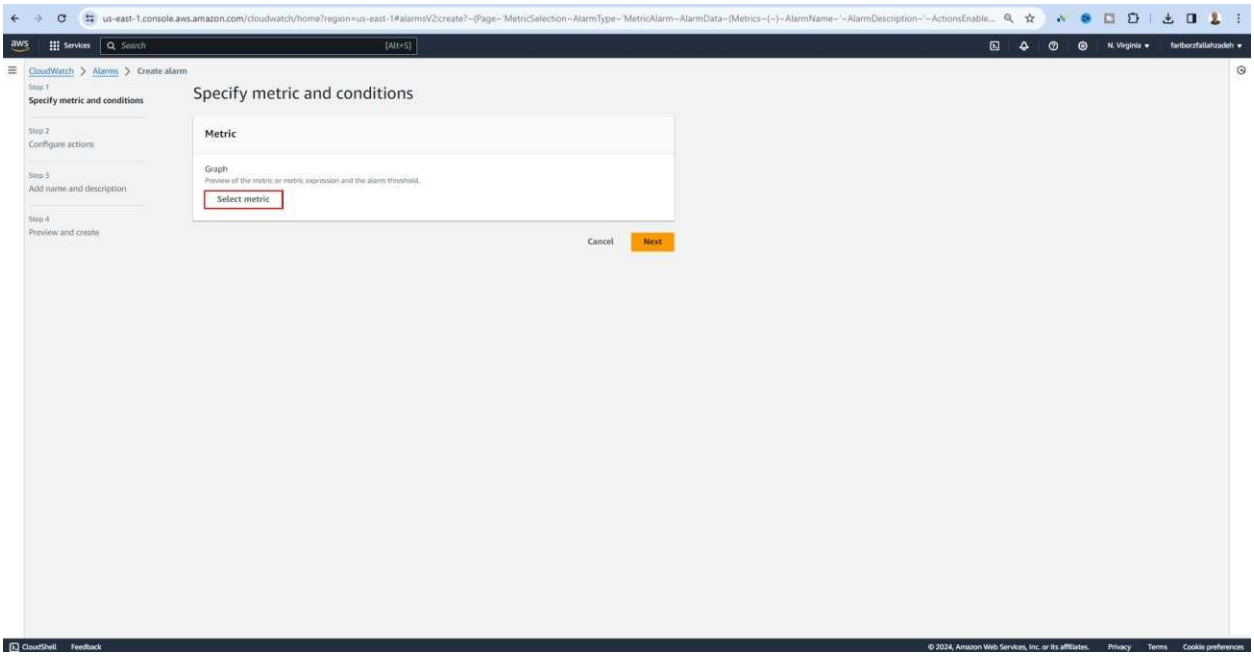
We need to be in a region called **North Virginia**. The pricing for services may vary across different regions, and this region is suitable for a Free Tier account.



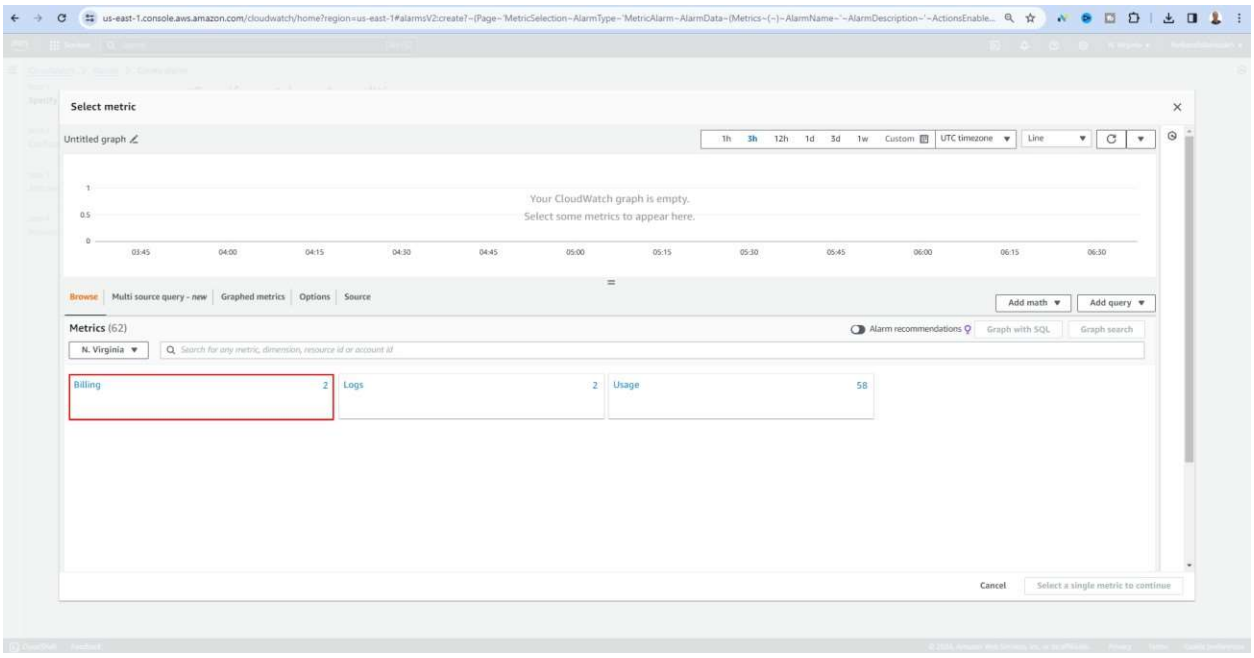
At this stage, click on the **All Alarms** link, then click the **Create Alarm** button.



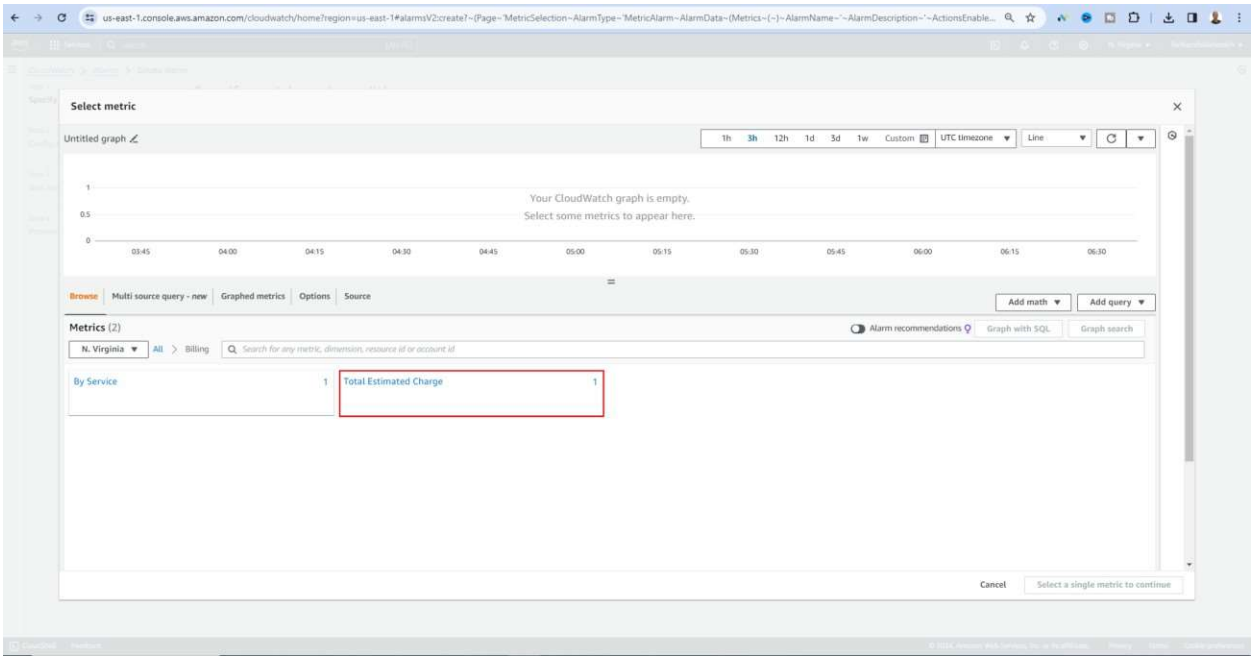
At this stage, click on the **Select Metric** button.



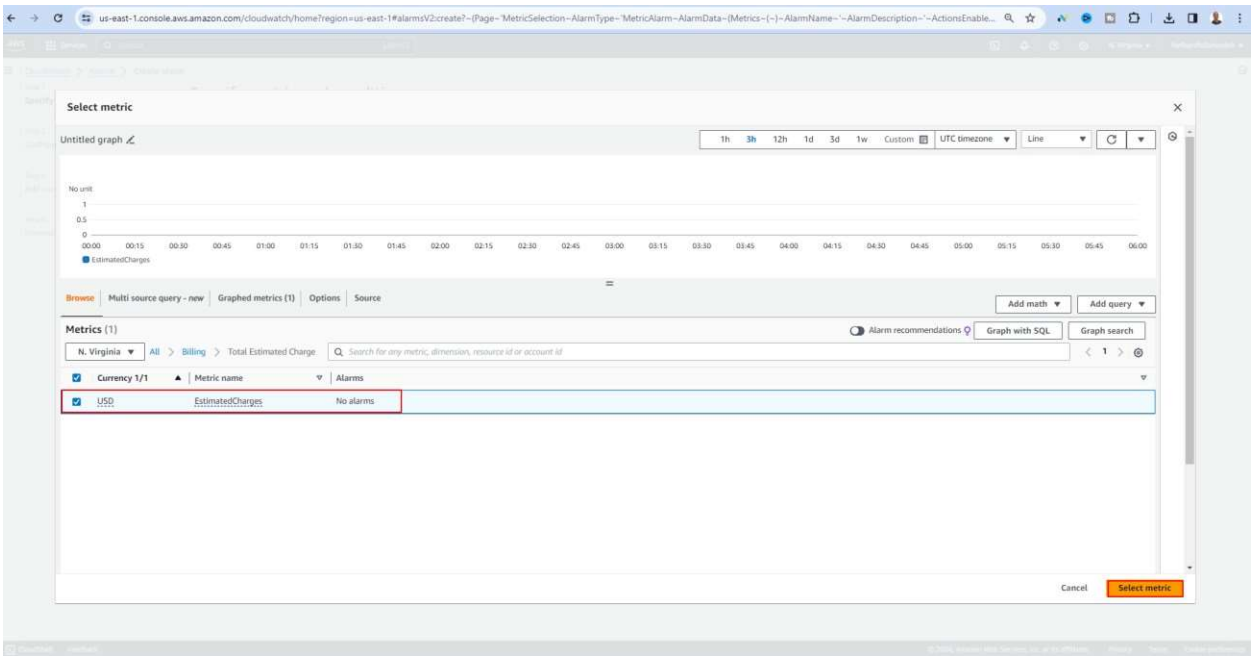
Click on the **Billing** link.



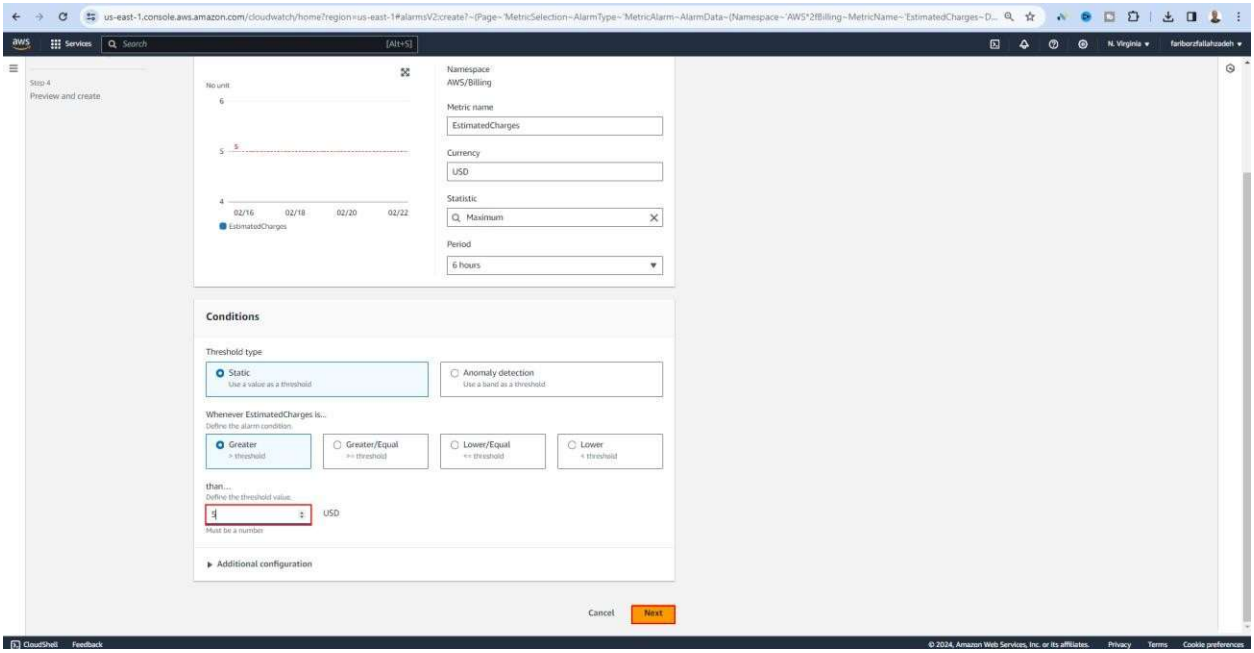
At this stage, click on the **Total Estimated Charge** link.



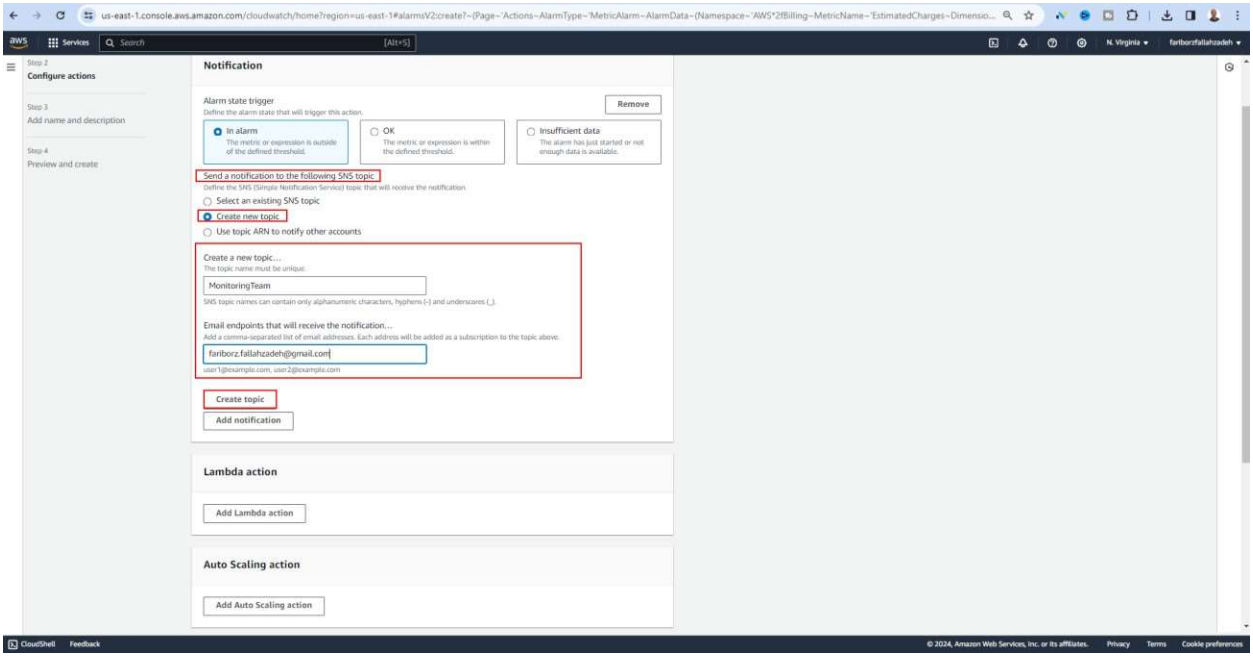
At this stage, select **USD Currency** and then click the **Select Metric** button.



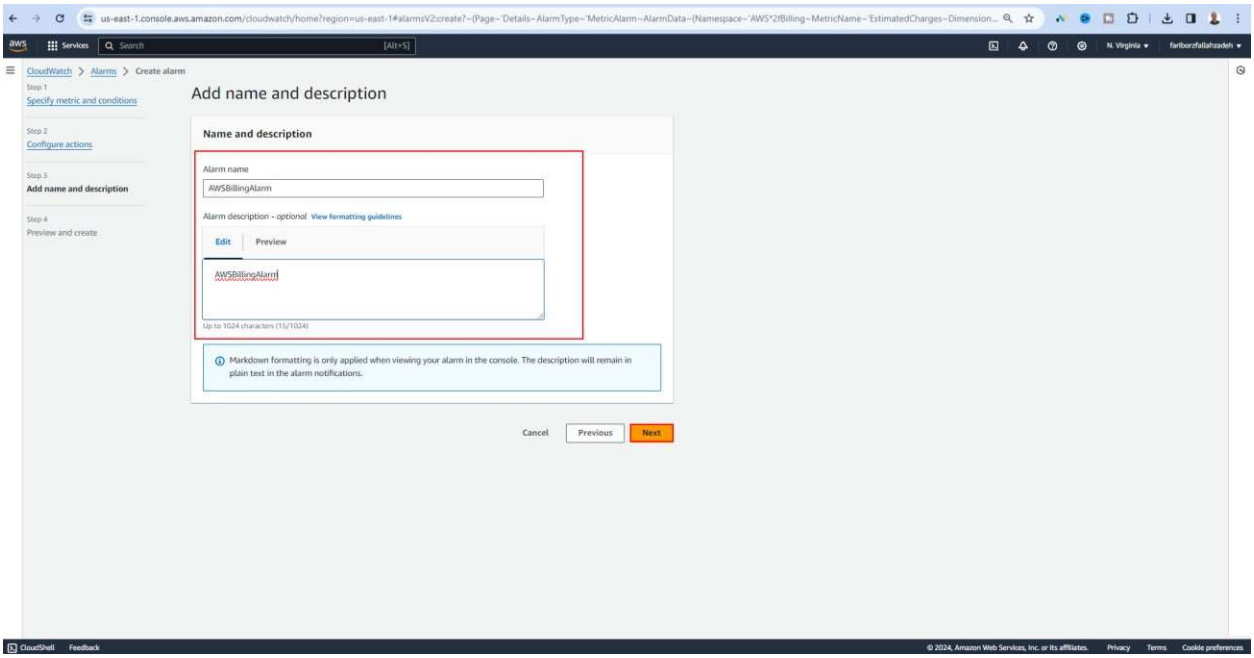
In the **Threshold value** section, set the value to **5 dollars** so that if the charges exceed 5 dollars, you will receive a notification via email. Then, click the **Next** button.



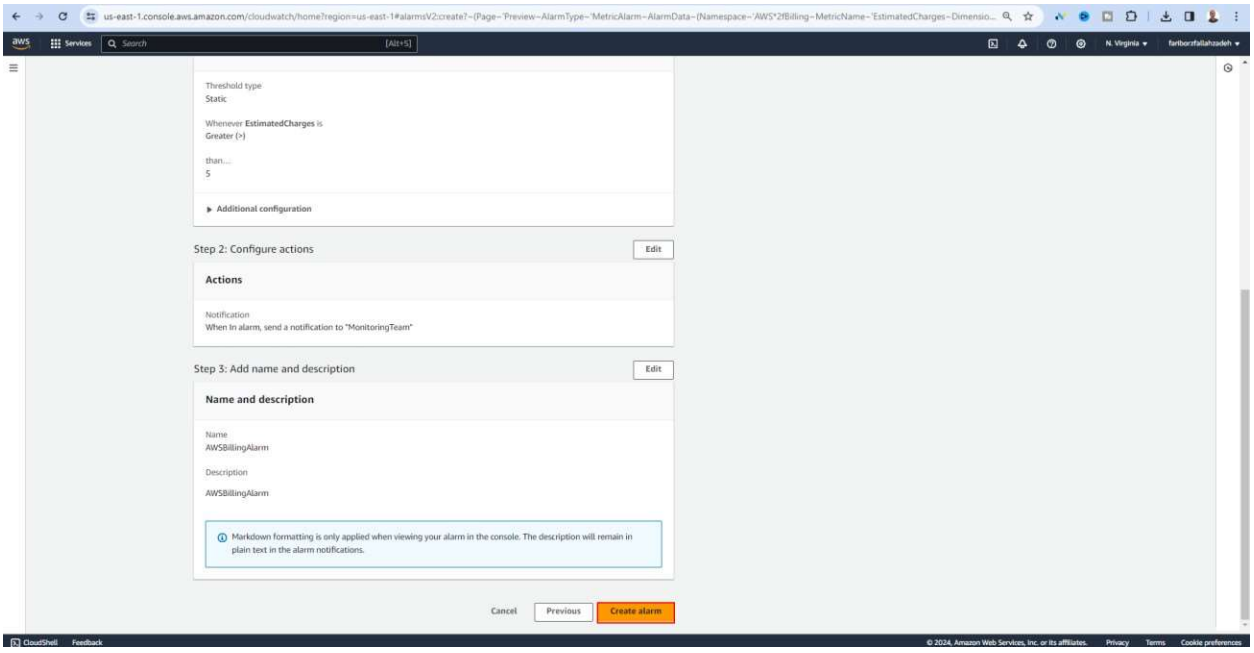
At this stage, you need to use a service called **SNS (Simple Notification Service)**. In this section, you will define an **SNS Topic**, which means you need to associate an email address with a topic. Define a name for the topic, then click the **Create Topic** button. Finally, click the **Next** button.



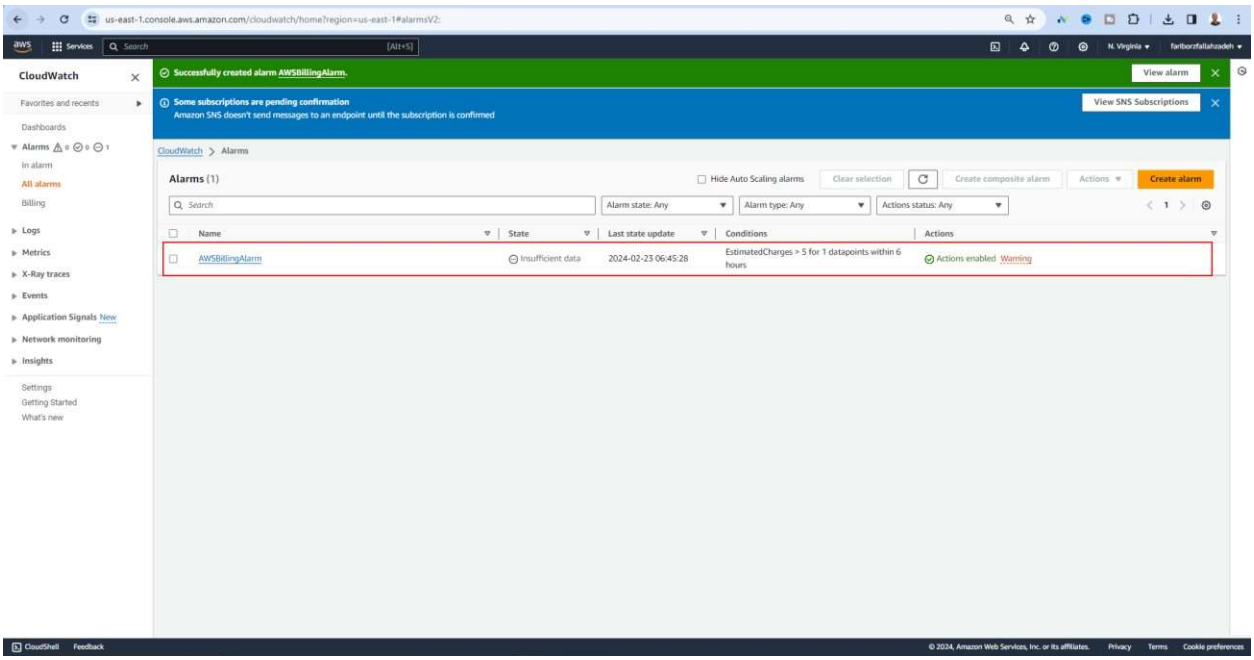
At this stage, you need to define an **Alarm Name** and an **Alarm Description**, then click the **Next** button.



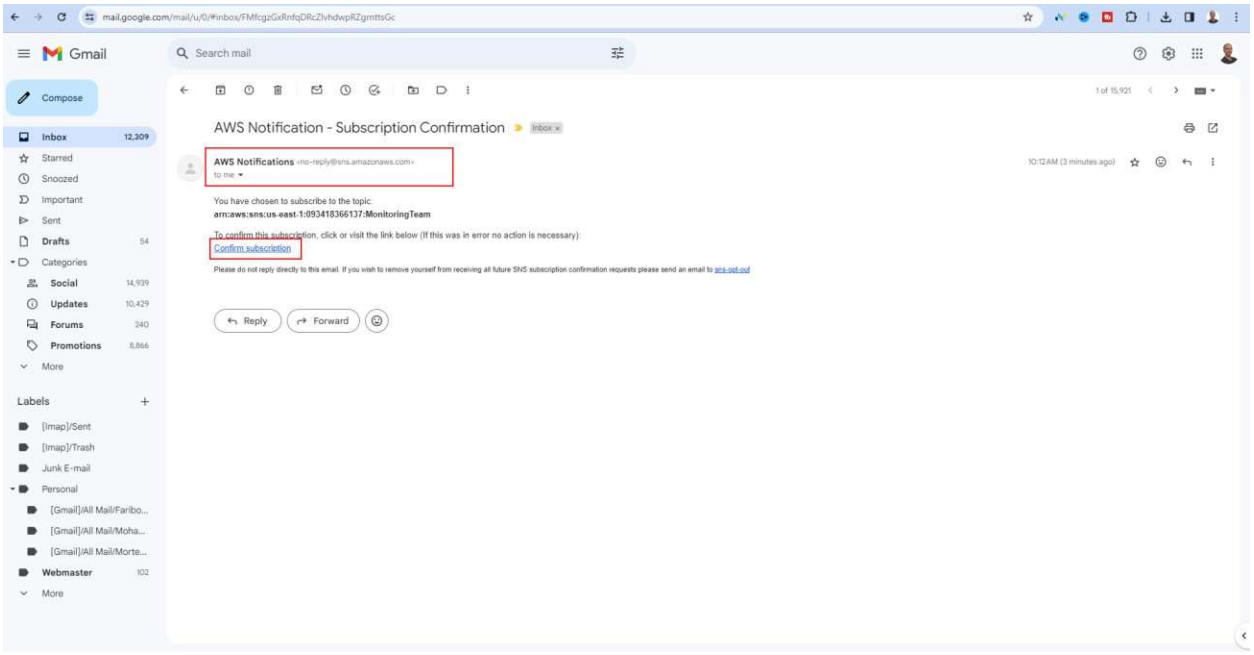
At this stage, click the **Create Alarm** button.



At this stage, we need to verify your **email address** to receive the alarm notifications.



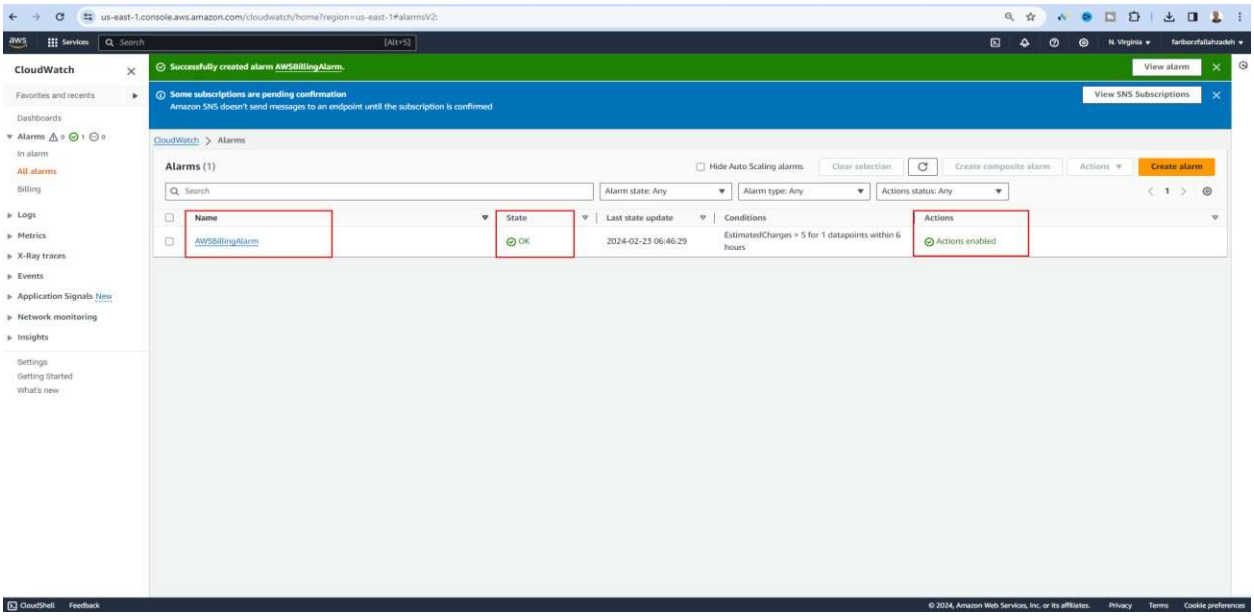
Go to your email address and click on the **Confirm Subscription** link.



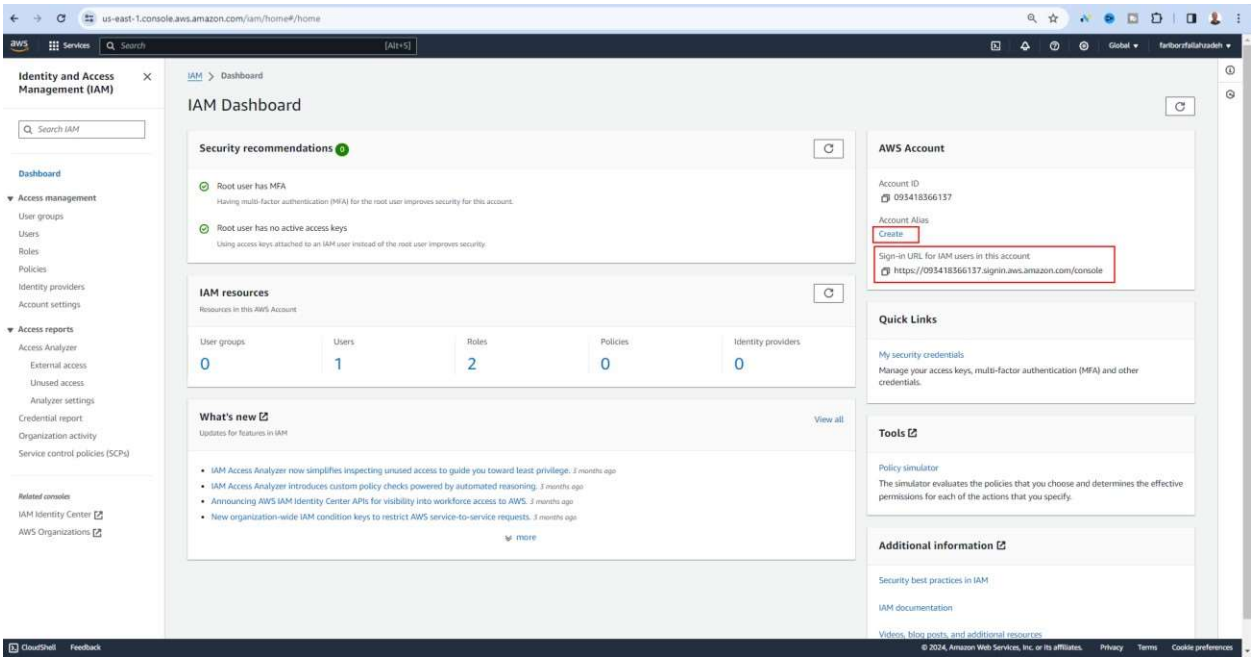
As shown in the image below, the email address has been successfully verified.

Next, return to the **CloudWatch Alarm** page and refresh the page. The alarm status should show as **OK**, and its action should be marked as **Action Enabled**.

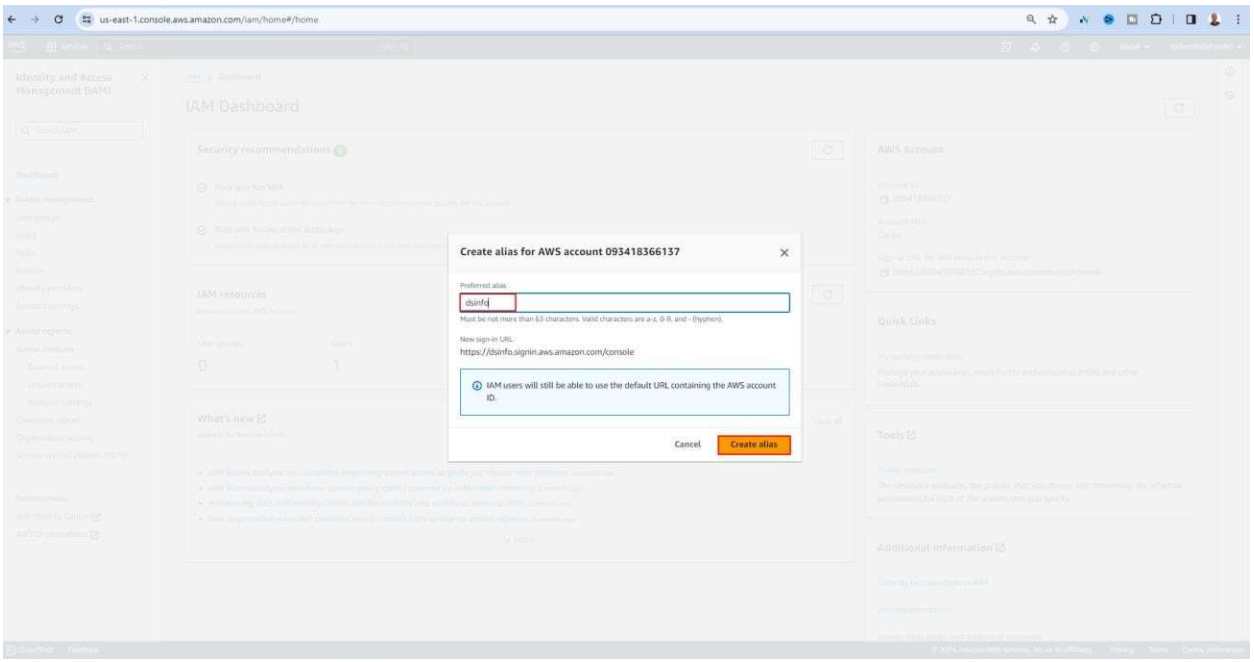
Currently, since the service usage is less than 5 dollars, the alarm status is shown as **OK**.



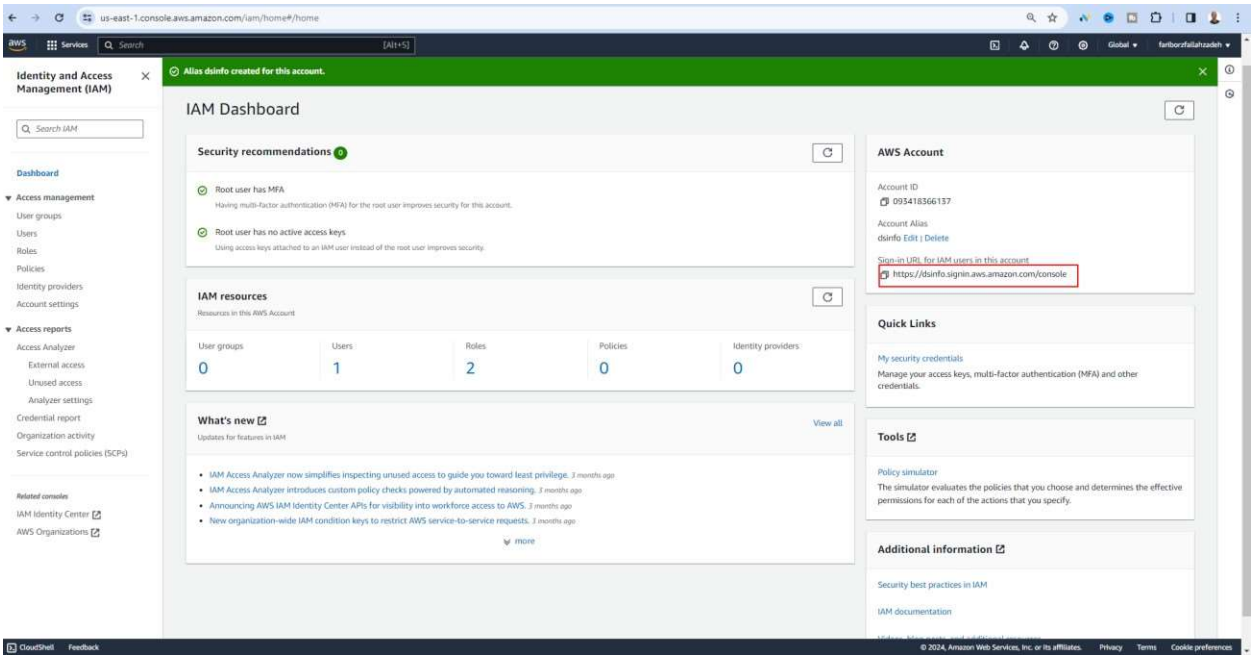
All the AWS setup steps have been completed, and now you can log in to the **AWS Management Console** using the **IAM User** through the link shown in the image. You can also set up an alias for this link.



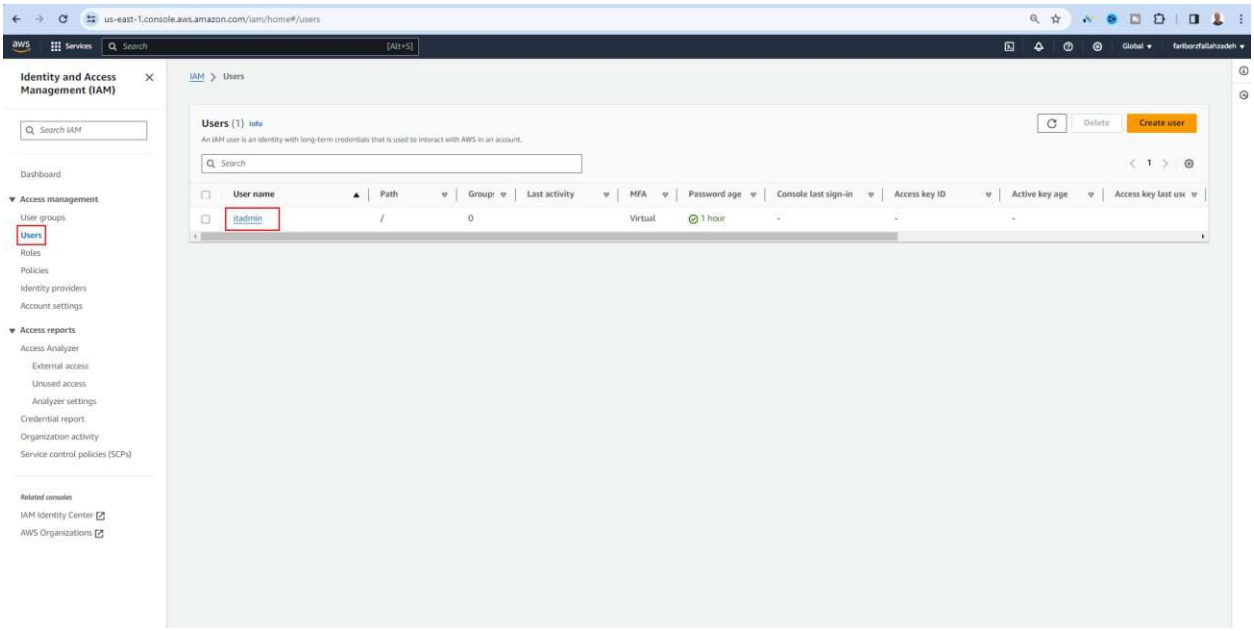
At this stage, define an **Alias**.



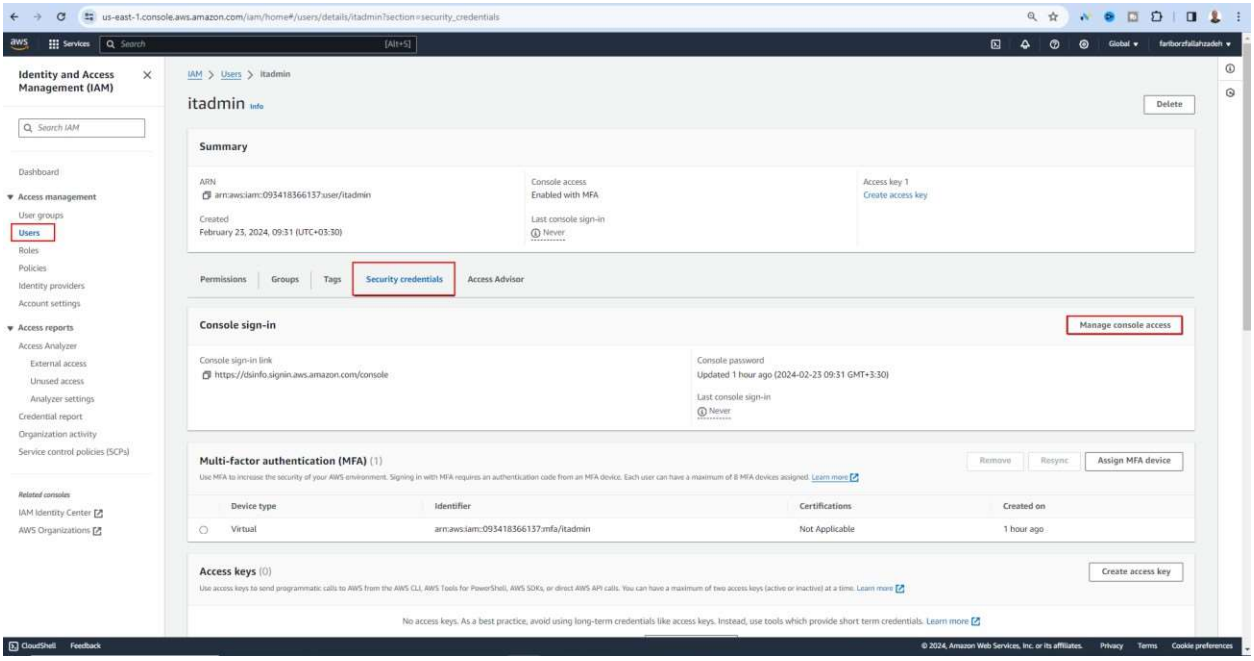
As shown in the image below, the alias for the link has been set up.



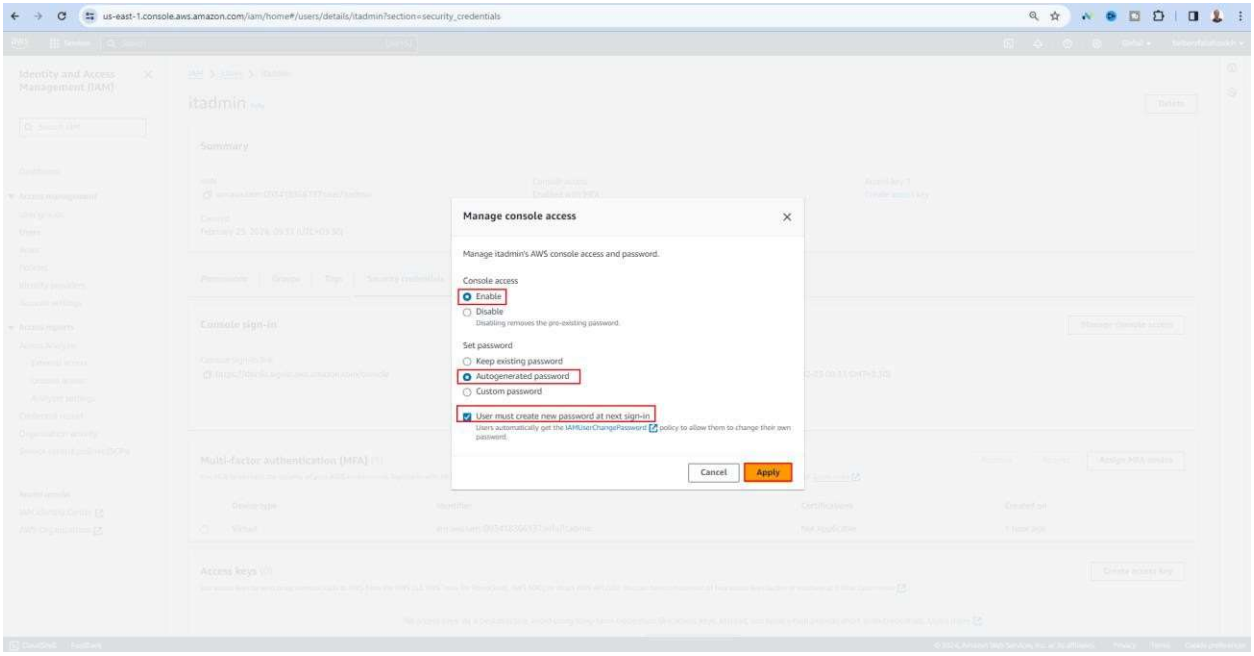
At this stage, click on the **Users** section, and then click on **IAM User**.



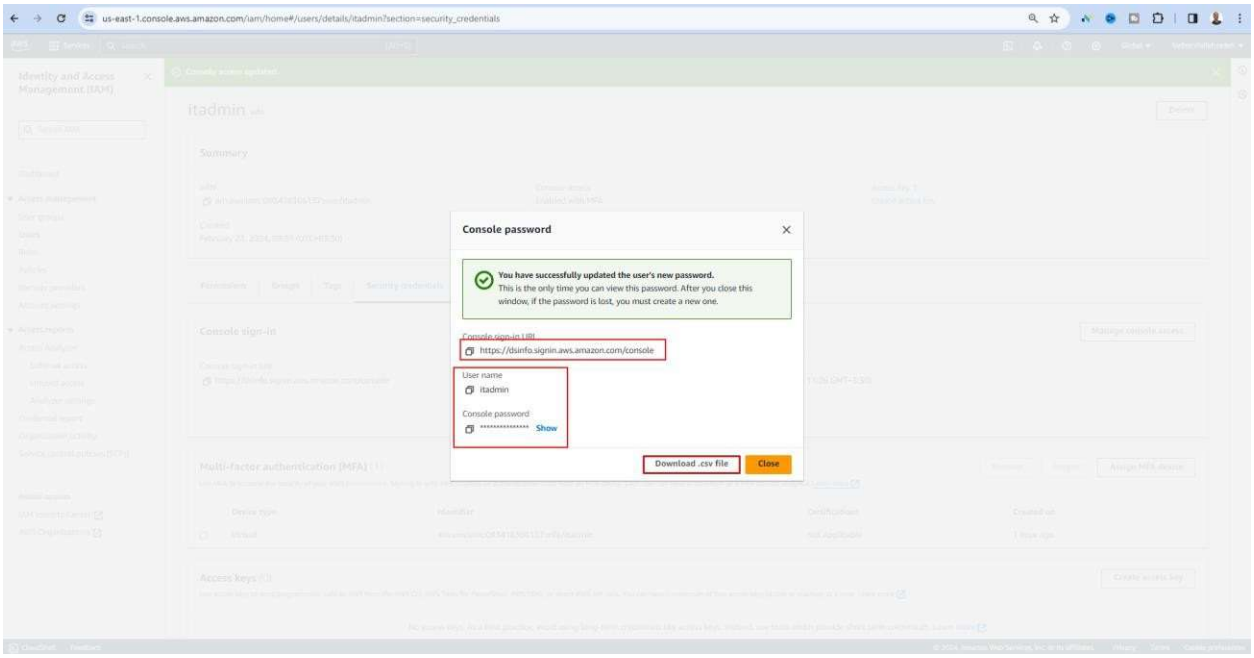
In the **Security Credentials** section, click on the **Manage Console Access** button.



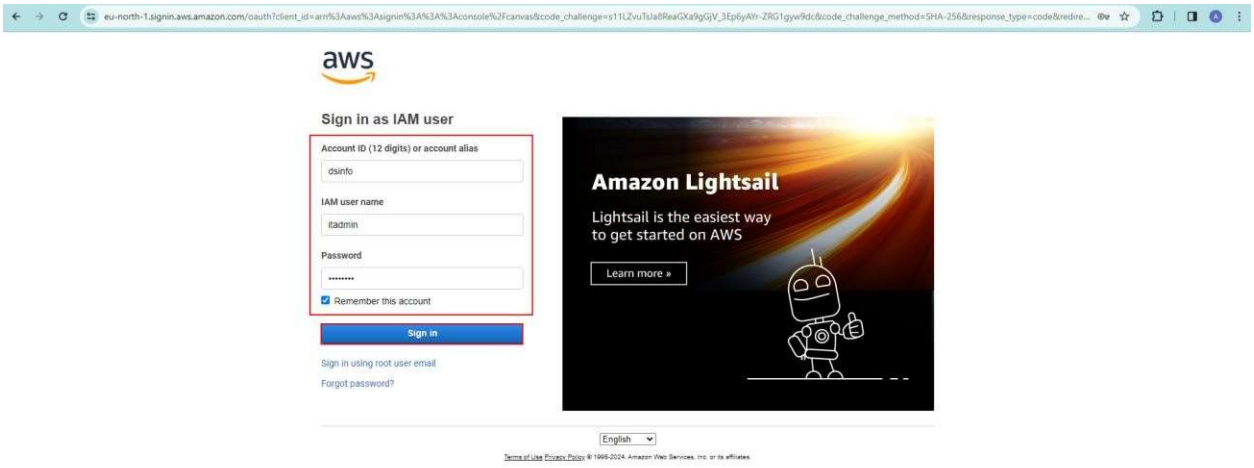
At this stage, select the following options and then click the **Apply** button.



At this stage, download the **CSV file**. This file contains the **URL**, **IAM User**, and **IAM Password**.

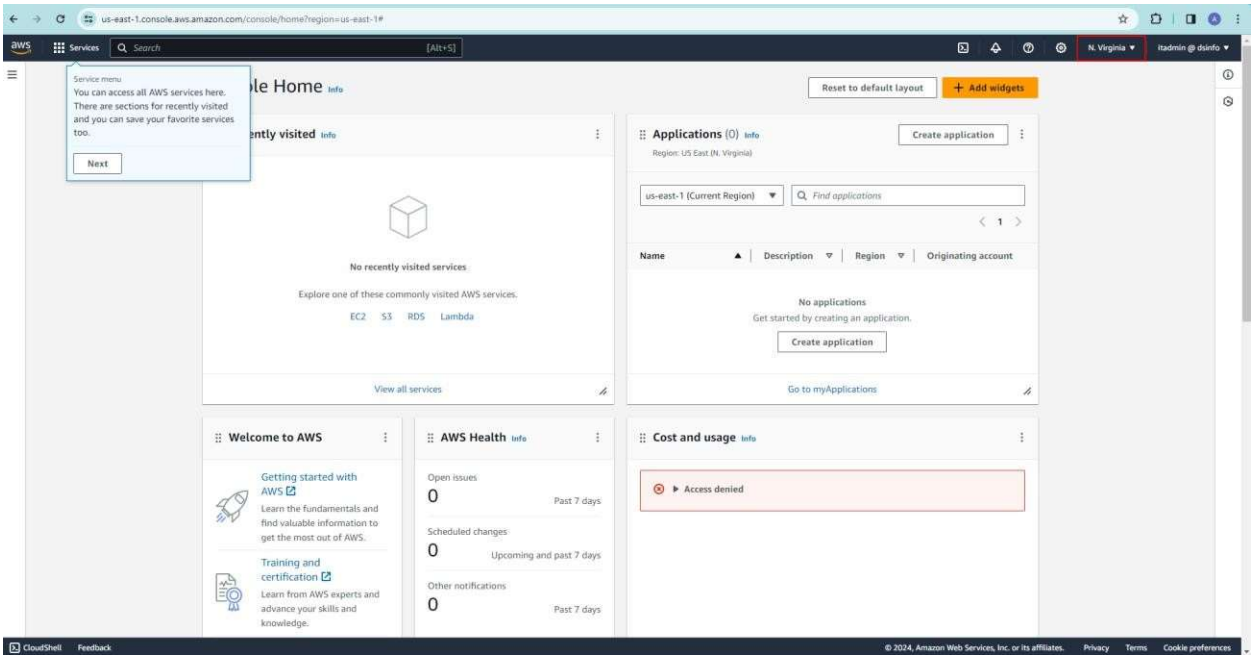


Then, go to the **AWS Management Console** link and sign in using the **IAM User** and **IAM Password** from the CSV file.



Since we have enabled MFA for the IAM user, you will need to enter the code generated by the **Google Authenticator** app in this section, and then click the **Submit** button.

As you can see, we have successfully logged in to the **AWS Management Console**.



Introduction to AWS Regions and Availability Zones

AWS is spread across the world and is present in most countries, continuously expanding. These countries are referred to as **Regions**.

Within each Region, there are several **Availability Zones**. Each Zone is like a **Data Center**, and an Availability Zone consists of multiple **Data Centers**.

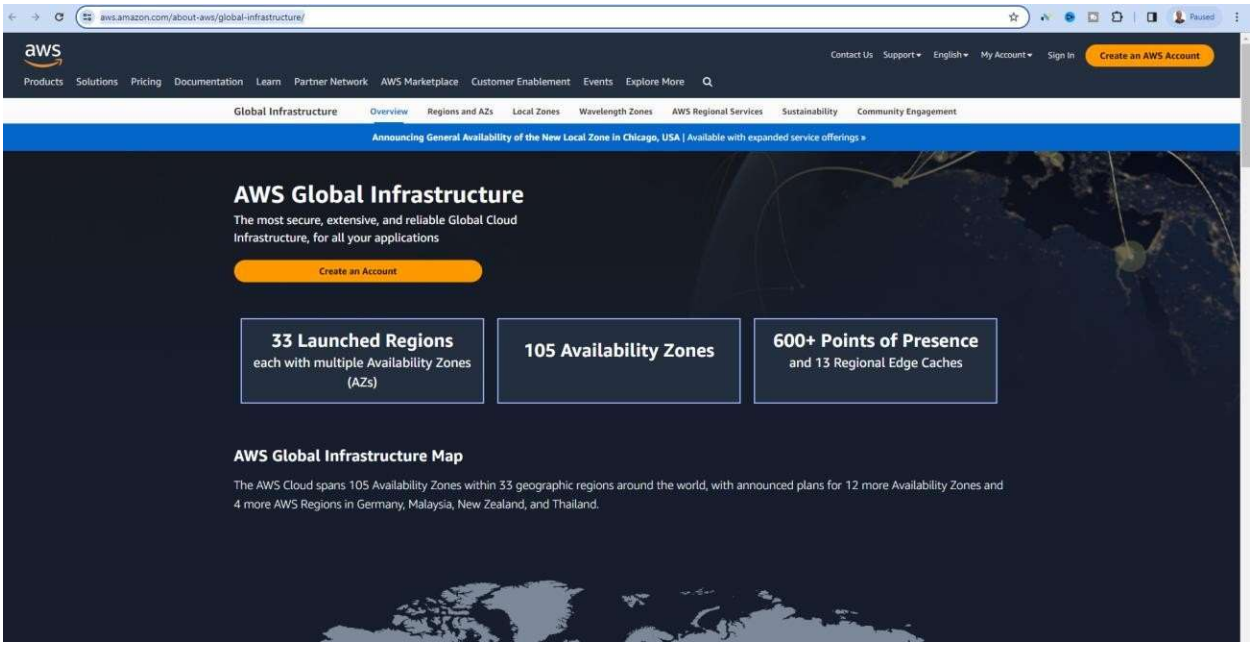
Each Region must have at least two **Availability Zones**.

AWS claims to have high and extensive security, and it is a reliable cloud platform. It offers more than **175 services** through its data centers, and these services are continuously increasing.

Currently, AWS has more than **105 Availability Zones** across **33 Geographic Regions** worldwide.

To understand the AWS Global Infrastructure, you can view the following link:

https://aws.amazon.com/about-aws/global-infrastructure/



By using these Availability Zones, you can achieve **High Availability** for your infrastructure. For example, if you have four web servers, you can place two web servers in one Availability Zone and the other two in another Availability Zone.

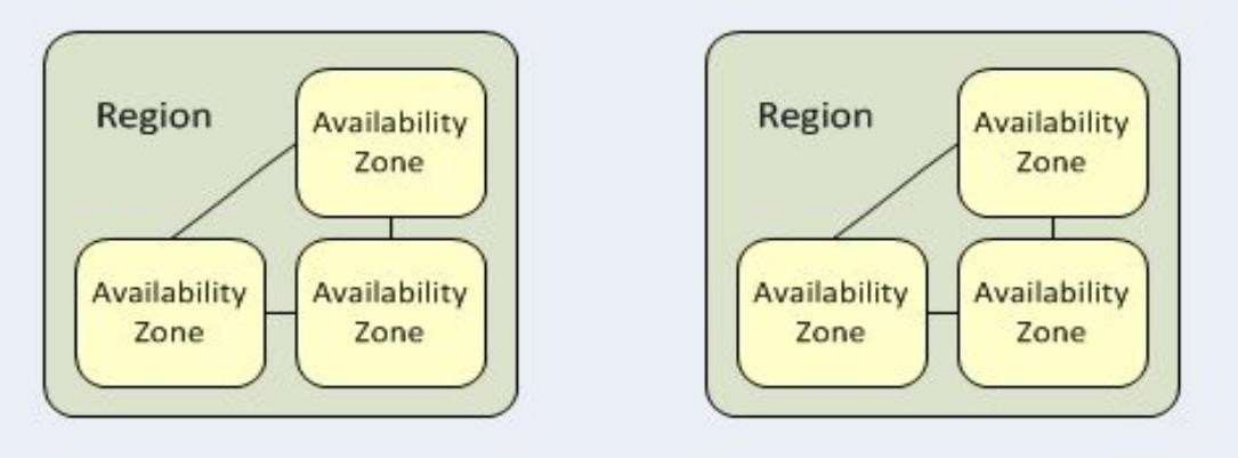
Therefore, if one of your Availability Zones goes down or becomes slow, your web server in the other Availability Zone will remain **up** and operational.

Benefits of Using AWS Infrastructure

- **High Availability** through multiple Availability Zones.
- **Improved Continuity** through replication across Regions.
- **Data Compliance** and required **Residency**.
- **Geographic Expansion** to leverage global infrastructure.

Introduction to Availability Zones

An **Availability Zone** means that you can have multiple zones, and you can distribute your infrastructure across several Availability Zones within your region to achieve **High Availability**.



For example, if you launch a **Virtual Machine**, which is known as an **Instance**, you need a place for it to reside, and that place is the **Availability Zone**. Alternatively, you can allow AWS to decide which Availability Zone your instance will be placed in.

When designing an application, for your **infrastructure layout**, you can choose multiple Availability Zones within a region, and your infrastructure and traffic will be distributed across these zones.

When you log in to the **AWS Management Console** using an **IAM User**, you can view all the services provided by AWS.

For example, if you are a **DevOps** specialist or a **SysAdmin**, you may need a virtual machine. You can use the **EC2** service from AWS to fulfill this need.

In this course, our focus will be on services like **EC2**, **Beanstalk**, and storage services such as **S3** or **EFS**, along with database services like **RDS** and **Elasticache**, which are relevant to **DevOps**, **SysAdmin**, and **Developer** roles.

In the **AWS Management Console**, you can select the **Region** for your project.

When you select a region, you can use the Availability Zones within that region, so your data will be stored across those zones.

When using AWS Free Tier, it's better to choose a **US-based Region** because other regions may have higher costs. Although the Free Tier account is free, it has limitations, and if you exceed those limits, you will incur charges.

Introduction to AWS EC2

EC2 (Elastic Compute Cloud) is one of the most popular AWS services. It provides you with virtual machines and related services.

Introduction to EC2 Features

-Web Service API

EC2 provides you with a **Web Service API** to manage, provision, and deprovision virtual machines within the cloud.

-Ease Scale Up/Down

You can easily perform **Scale Up** or **Scale Down** operations on your resources in EC2.

For example, if you have 8GB of RAM, you can easily upgrade it to 16GB (Scale Up), or you can reduce it to 4GB (Scale Down).

You can scale up or scale down all the resources of a virtual machine, such as **CPU**, **network**, **storage**, and more.

The ability to scale up or scale down resources is due to the **elasticity** of the EC2 service.

-Pay Only For What you User

You pay for what you use, and how much you consume is calculated using different methods.

-Can Be Integrated into Several Other Services

The EC2 service easily integrates with various other services like **S3**, **EFS**, and more.

Introduction to EC2 Pricing

The cost of using EC2 is categorized into the following four methods:

-On Demand

It is calculated on a per-second or per-hour basis, and you don't need to pay for the entire duration — it depends on the amount of time you use the service.

-Reserved

You can reserve capacity for **1 to 3 years** and take advantage of discounts on the service.

-Spot

Amazon EC2 **Spot Instances** are a type of compute instance in Amazon EC2 cloud services that allow users to access computing resources (such as CPU and memory) at a lower cost compared to reserved or default EC2 instances. Essentially, Spot Instances enable users to take advantage of **unused capacity** in Amazon's data centers, often referred to as "empty space," in a cost-effective manner.

These EC2 instances are offered through an auction system, meaning that users specify their **bid price** for each compute resource. If the current price set by Amazon matches the user’s bid, the Spot instance becomes available. However, if the user’s bid price exceeds the current price, the Spot instance will be terminated and then reactivated when the price matches the user's bid or when other compute resources become available.

-Dedicated Host

In this case, you have a **Dedicated Host**, but it comes at a higher cost.

In this course, we will be using **On Demand** and **Free Tier** accounts.

Introduction to EC2 Components

An EC2 service consists of the following components:

-AMI

When you want to launch an EC2 instance, you need an **AMI** (Amazon Machine Image). AWS provides a large number of AMIs for creating EC2 instances.

-Instance Type

When you launch an EC2 instance, the **Instance Type** specifies the hardware configuration that the EC2 instance will use.

For example, it defines the amount of **CPU**, **RAM**, and **network** resources required for the EC2 instance.

-Amazon Elastic Block Store

EBS, which stands for **Elastic Block Storage**, is a type of storage in AWS. Each AMI comes with a storage volume — for example, it could be **8GB** for a Linux machine or **30GB** for a Windows operating system.

Therefore, there are different types of **AMIs** that can use **EBS storage**.

You can specify the **storage size** or attach your own **storage** to the EC2 instance.

The **EBS service** is a **virtual hard disk** that can be used to store an operating system or data.

-TAG

For each resource in AWS, you have a **TAG**, which is used in a **Key:Value** format. A tag can be something like the **EC2 Name**, **Project**, or the **customer** who will use your service. You can define and use multiple tags as needed.

Defining **tags** can be very useful for **filtering** and **billing** purposes.

-Security Group

A **Security Group** acts as a **virtual firewall** used to control traffic for one or more instances.

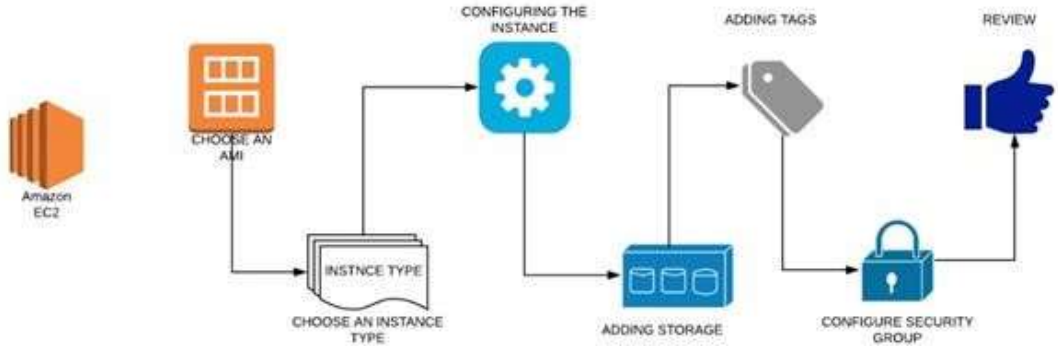
-EC2 Public Key

An **EC2 instance** uses **public key cryptography** to encrypt or decrypt login information.

Introduction to EC2 Instance Creation Steps

Creating an EC2 instance is very simple and **wizard-based**. You can select the **AMI**, choose the **Instance Type**, and configure other settings as needed.

In the image below, you can see the **steps for creating an EC2 instance**.



How to Create an EC2 Instance

In this section, we will learn **how to create an EC2 instance**.

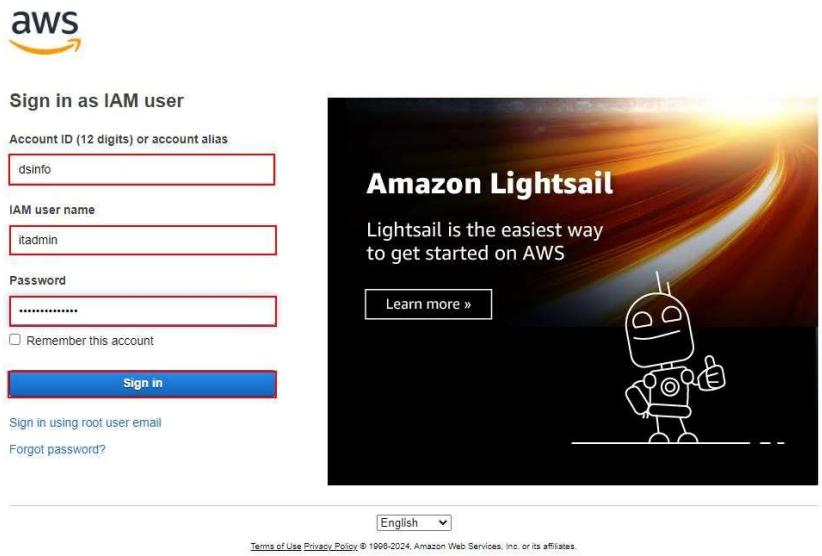
The **EC2 Instance** service refers to a **virtual machine**.

To create an EC2 instance, you need to follow these steps:

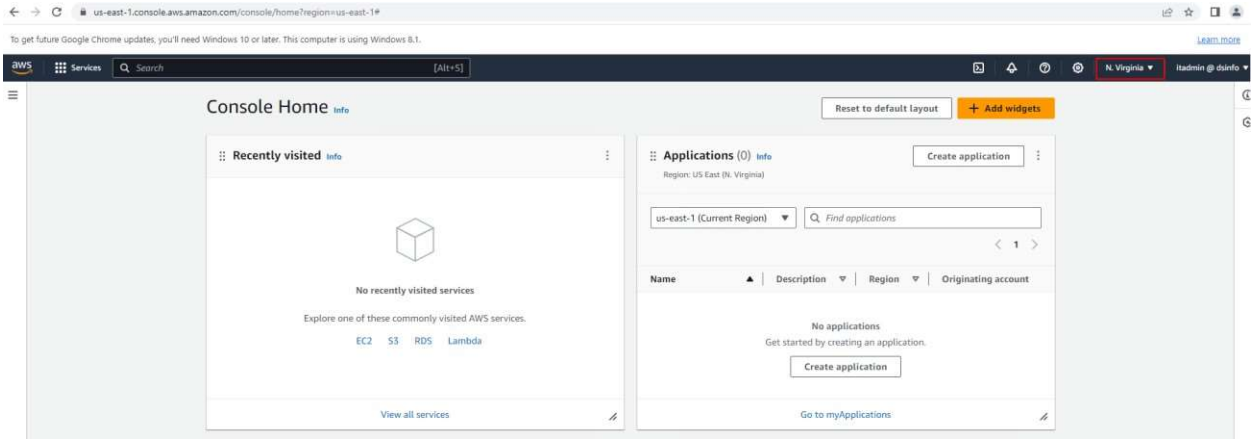
Step 1:

At this stage, you need to log in to the **AWS Management Console** using your **IAM User** and select your desired **Region**.

Logging in to AWS Management Console with IAM User

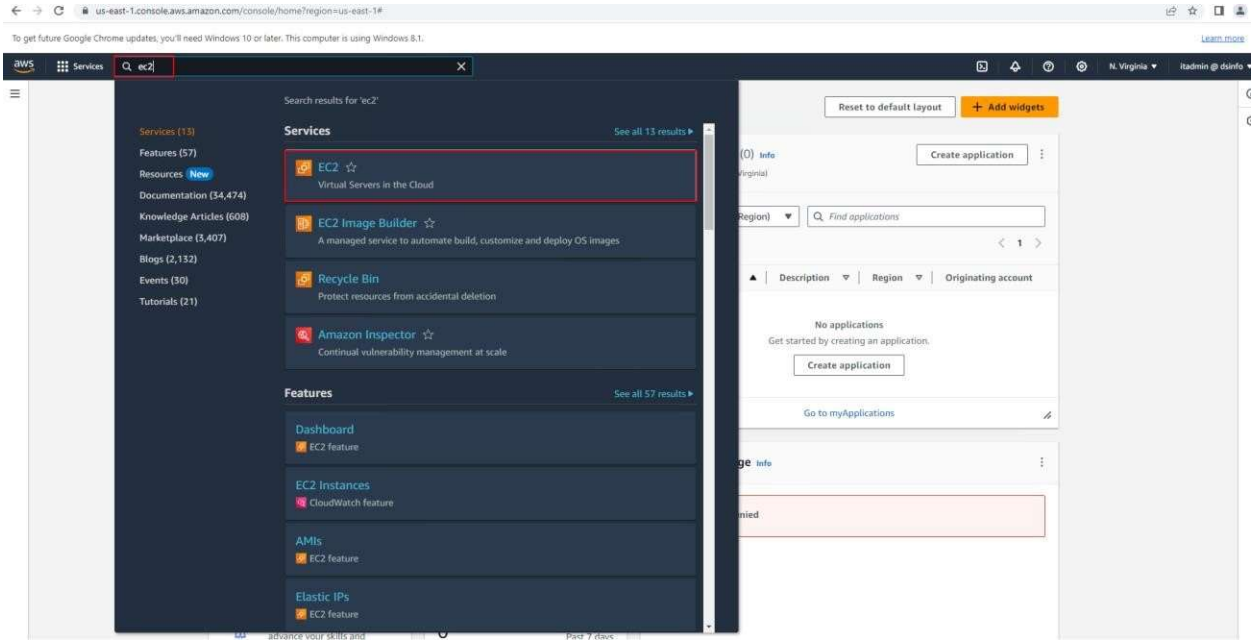


Selecting a Region



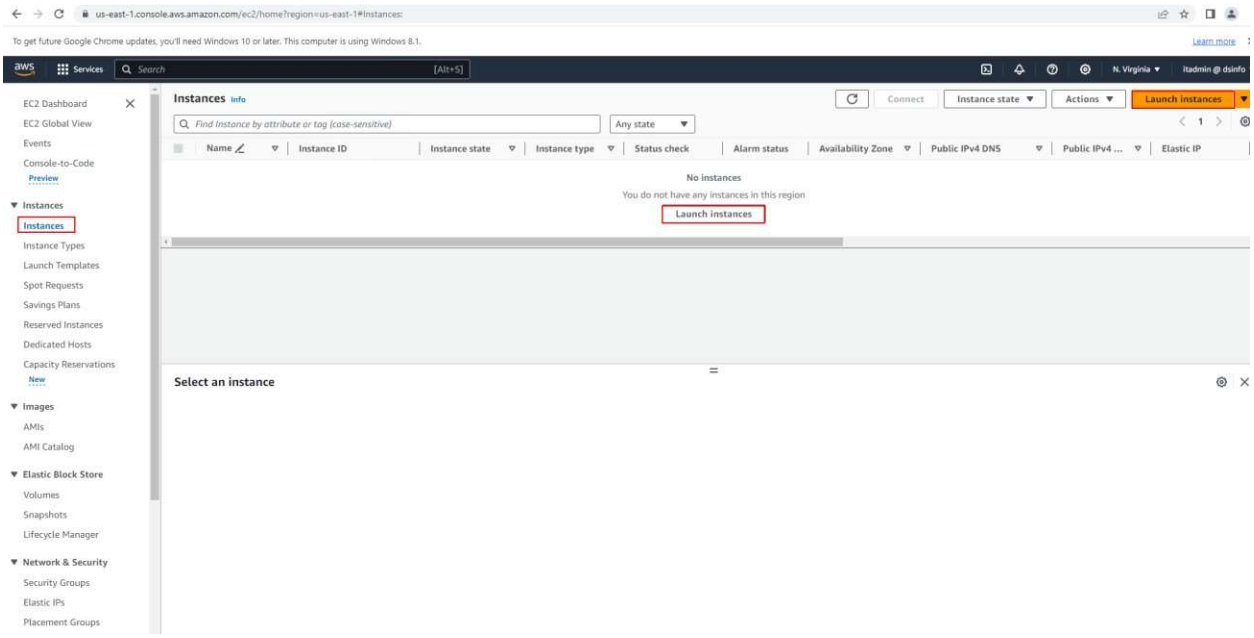
Step 2:

Selecting the EC2 Instance Service



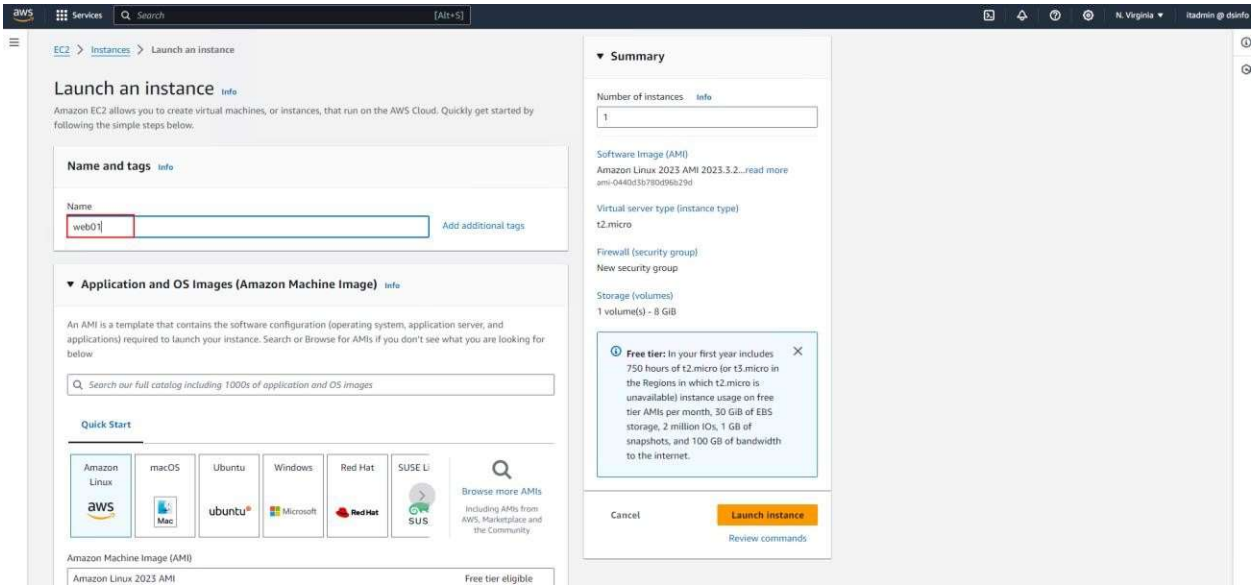
Step 3:

Click on the **Instances** link, then click the **Launch Instance** button.



Step 4:

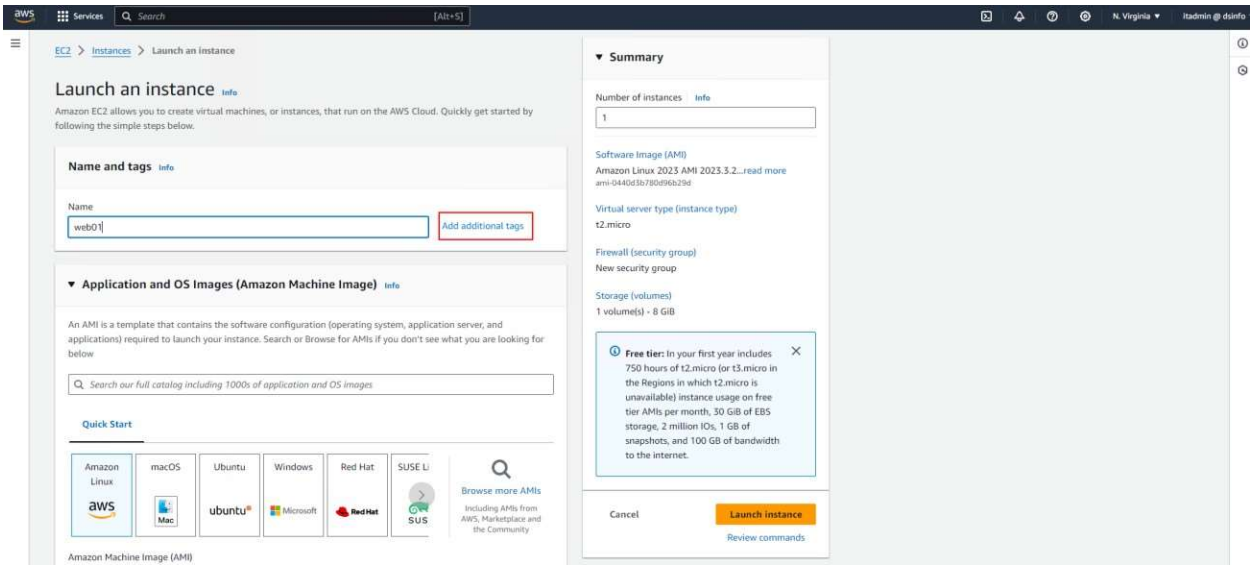
Specify a name for the instance.



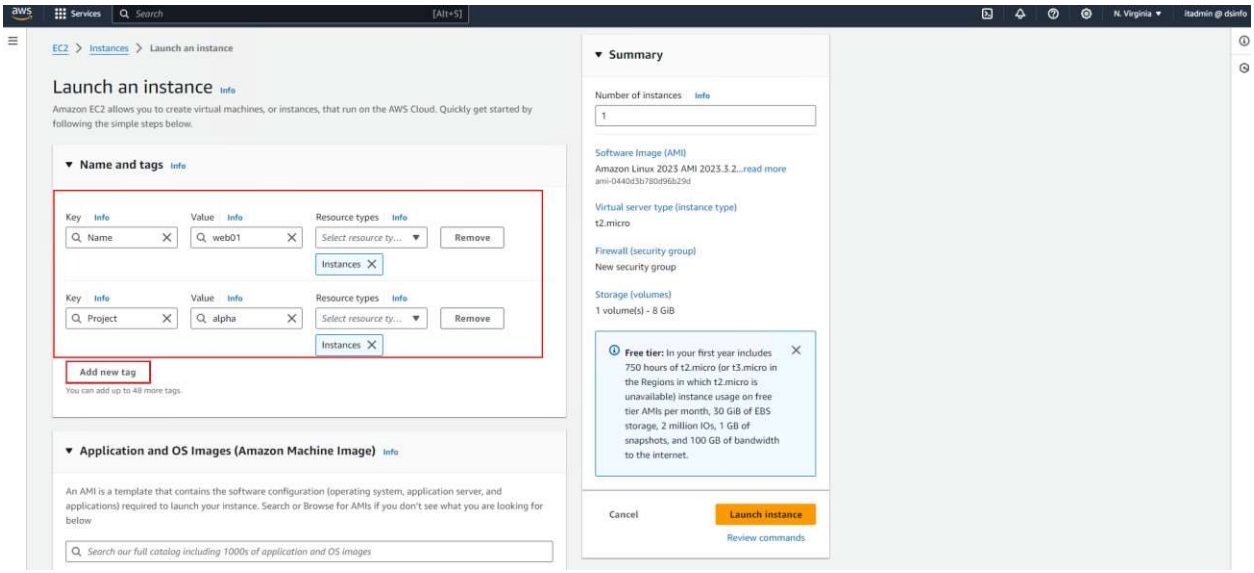
Step 5:

Specify a **TAG** for the instance.

To define a tag, click on the **Add Additional Tag** link.

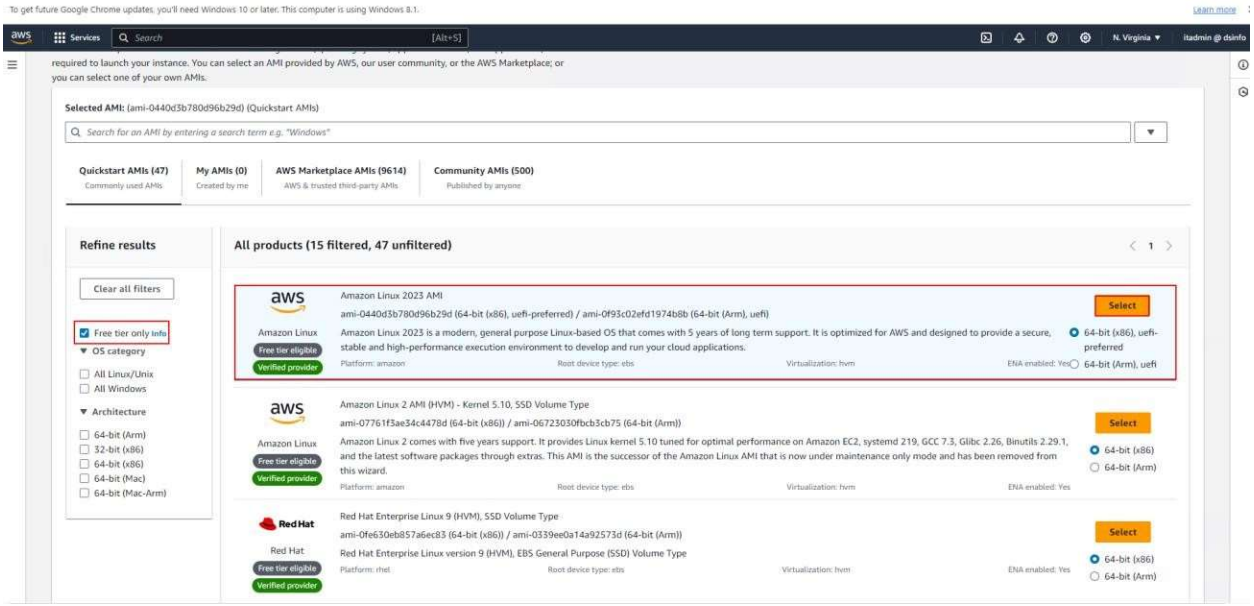


In this section, you can define a **TAG** for your instance.



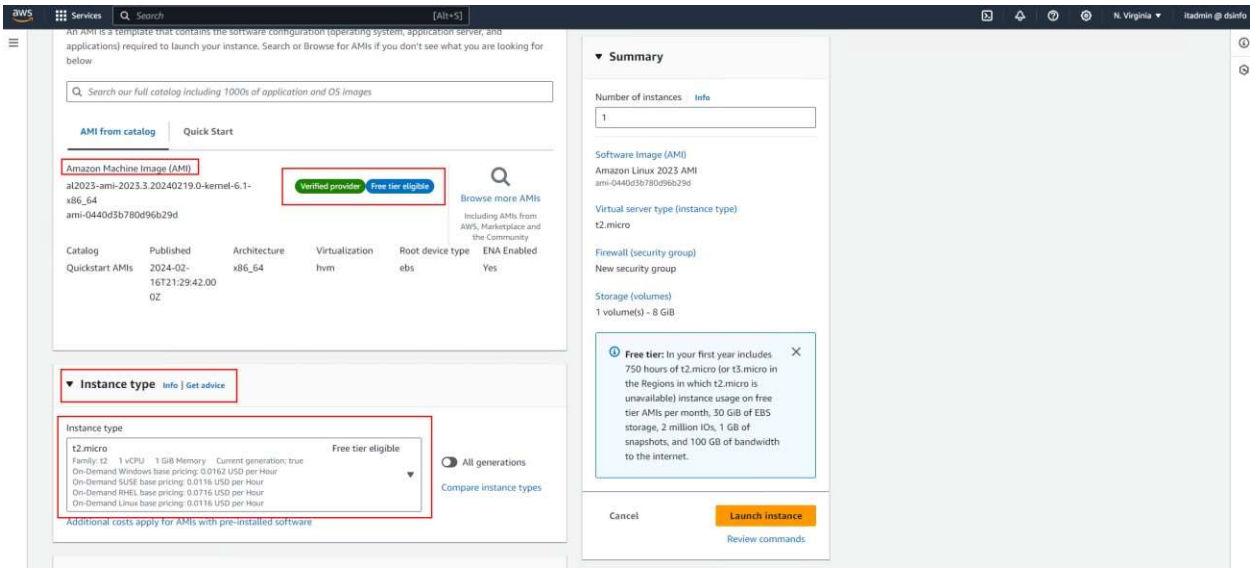
Step 6:

Specify the **AMI (Amazon Machine Image)**.



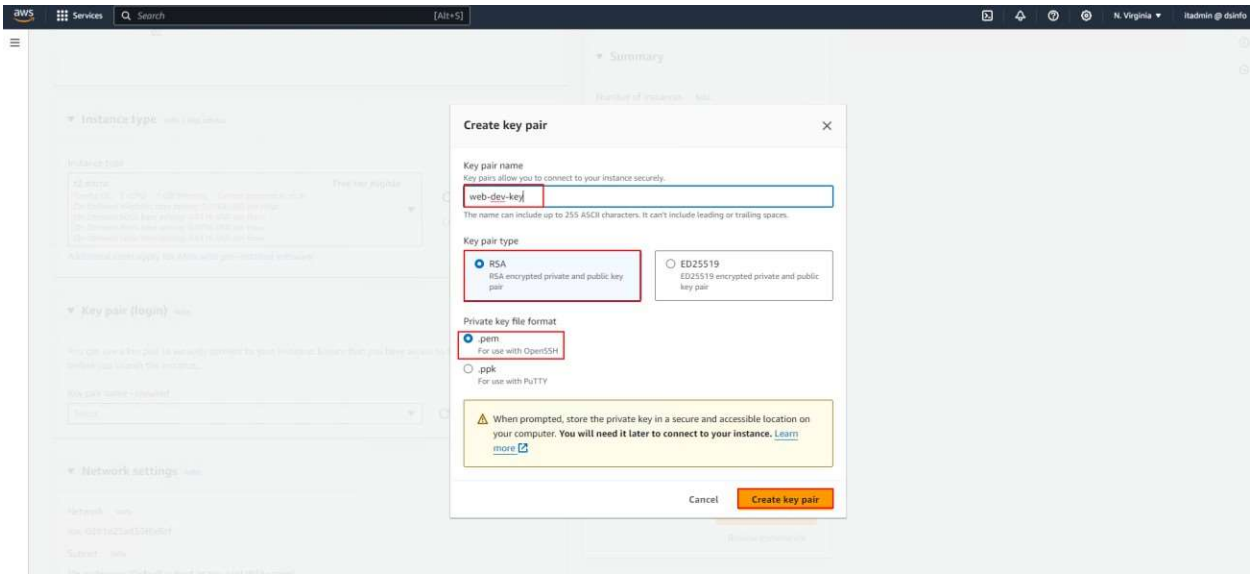
Step 7:

Select the **Instance Type**.



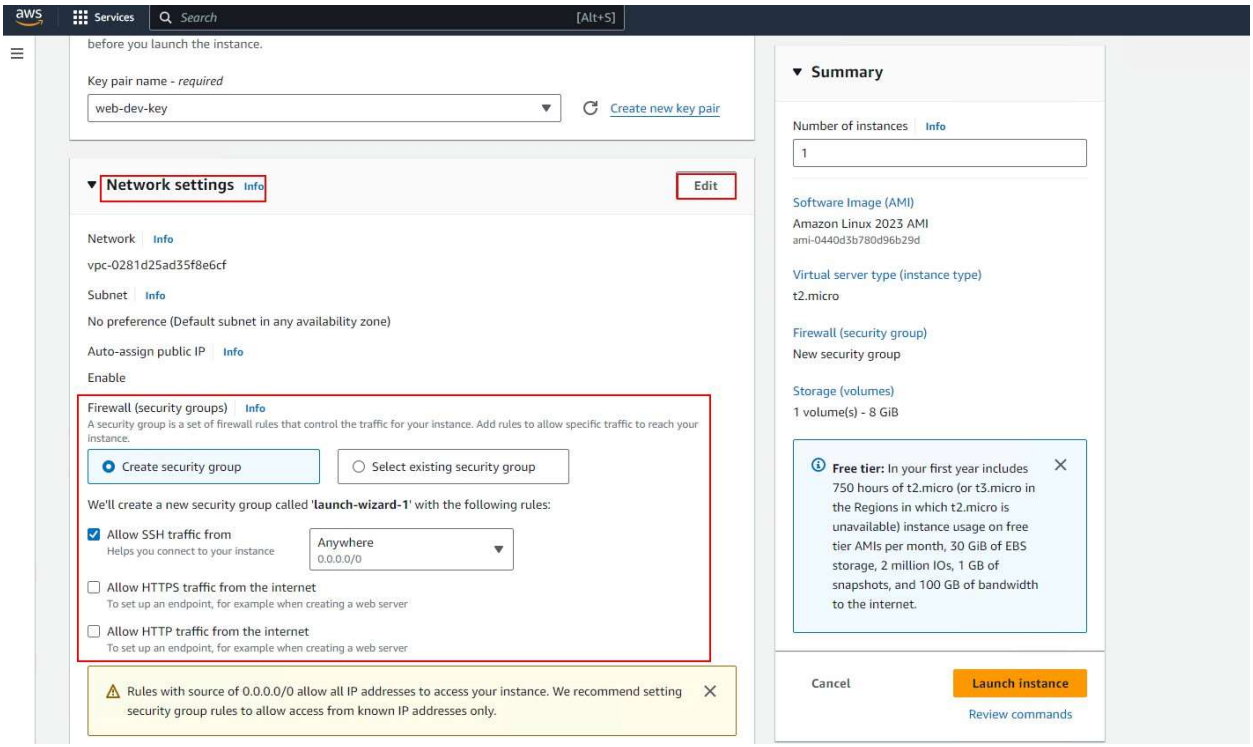
Step 8:

At this stage, you need to define a **Key Pair** for logging in to the EC2 instance.



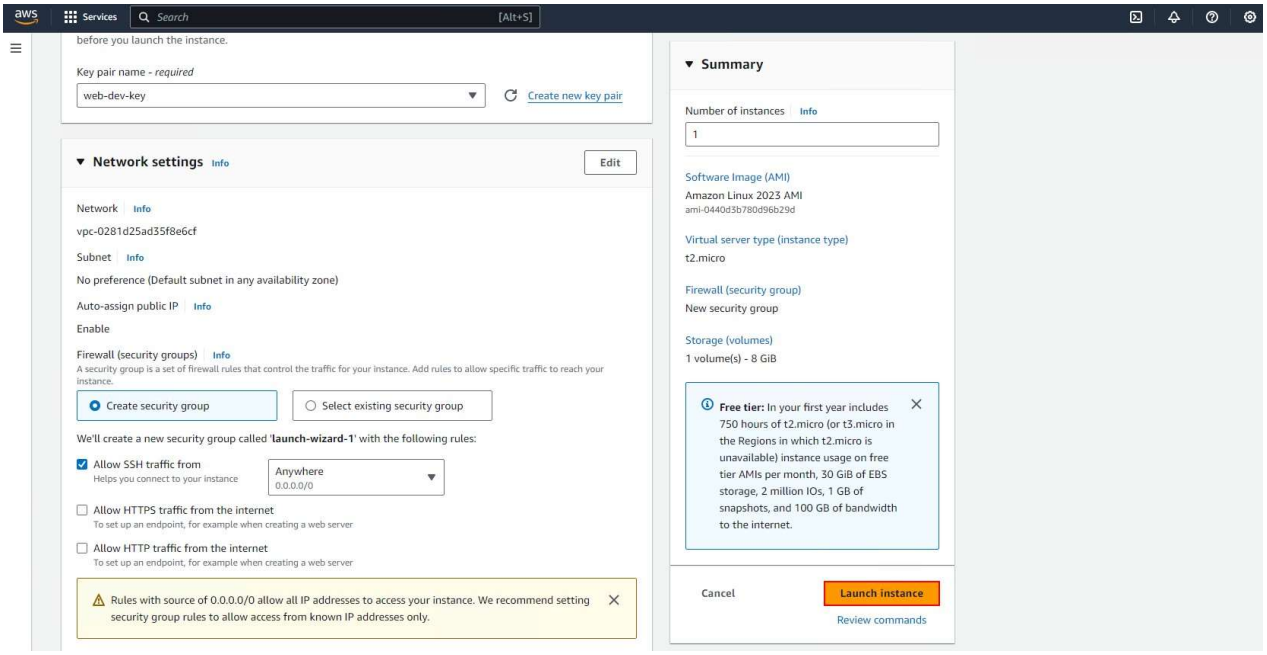
Step 9:

In this section, you can configure the **Security Group** settings.

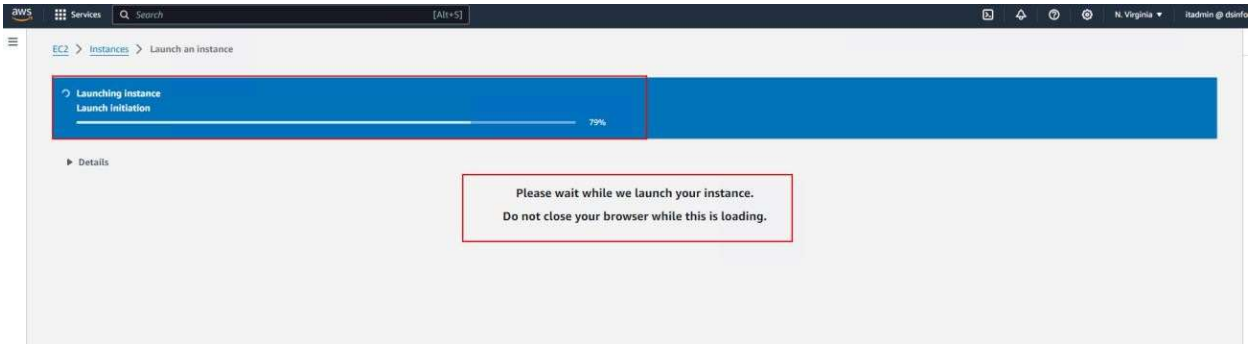


Step 10:

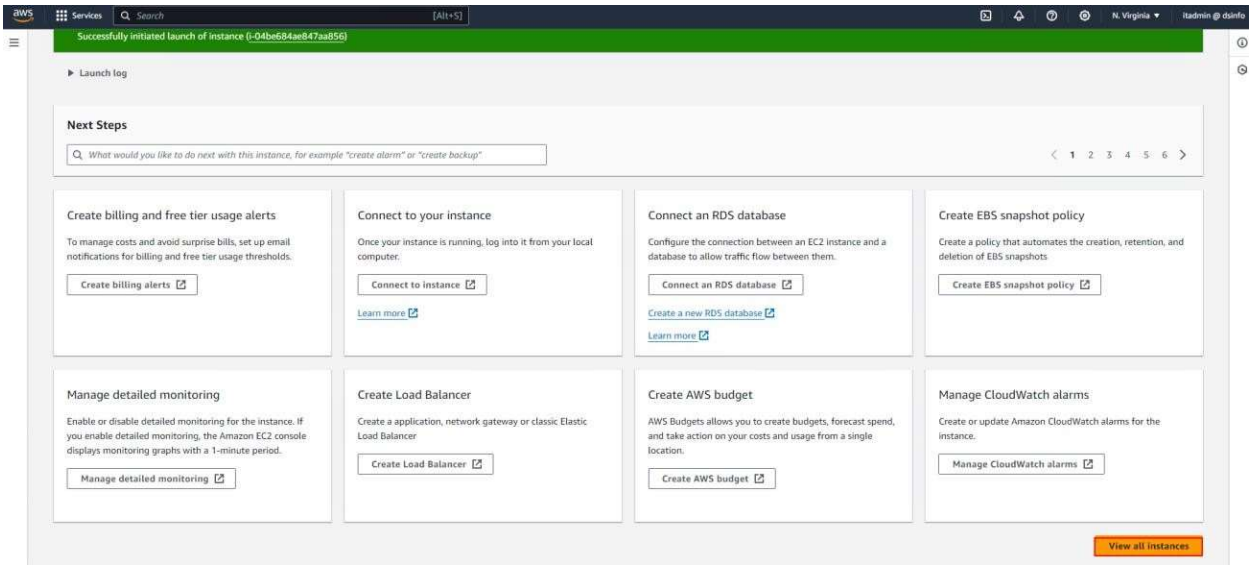
At this stage, click the **Launch Instance** button to create the virtual machine.



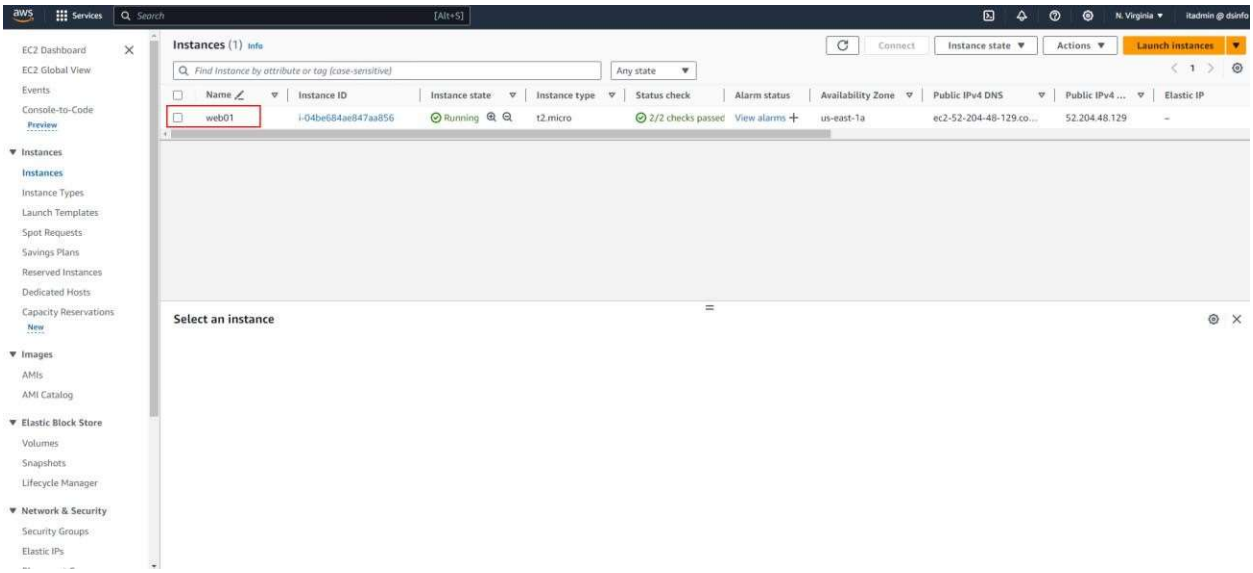
As shown in the image below, the **EC2 instance** is being created.



To view all instances, click on the **View all Instances** button.



As you can see, the instance has been successfully created.



Introduction to EC2 Best Practices

Before launching an EC2 instance, it's a good idea to become familiar with its **best practices**.

When you launch an EC2 instance, you need to have certain **requirements** in place, which include the following:

-Requirement Gathering

Defining the **task** — for example, setting up a web server, specifying the type of operating system, storage size, and compute size.

-Key Pair

Creating a **Key Pair**, which can be done either before or during the instance creation process.

-Security Group

Creating a **Security Group**, which acts as a firewall.

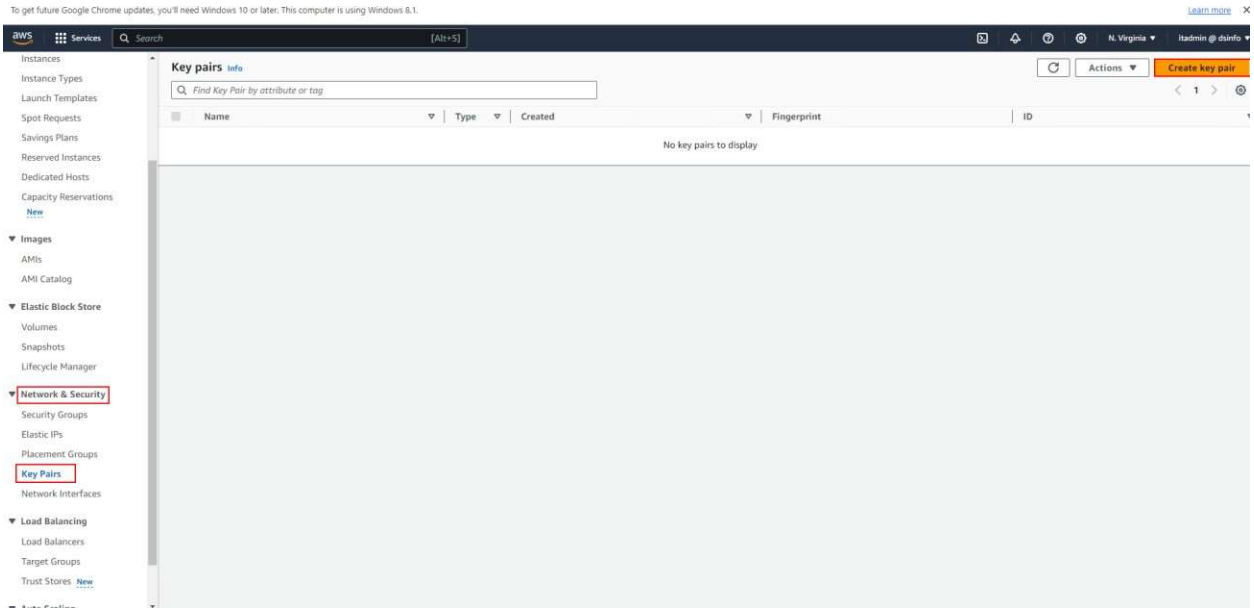
-Instance Launch

Launching the **Instance**.

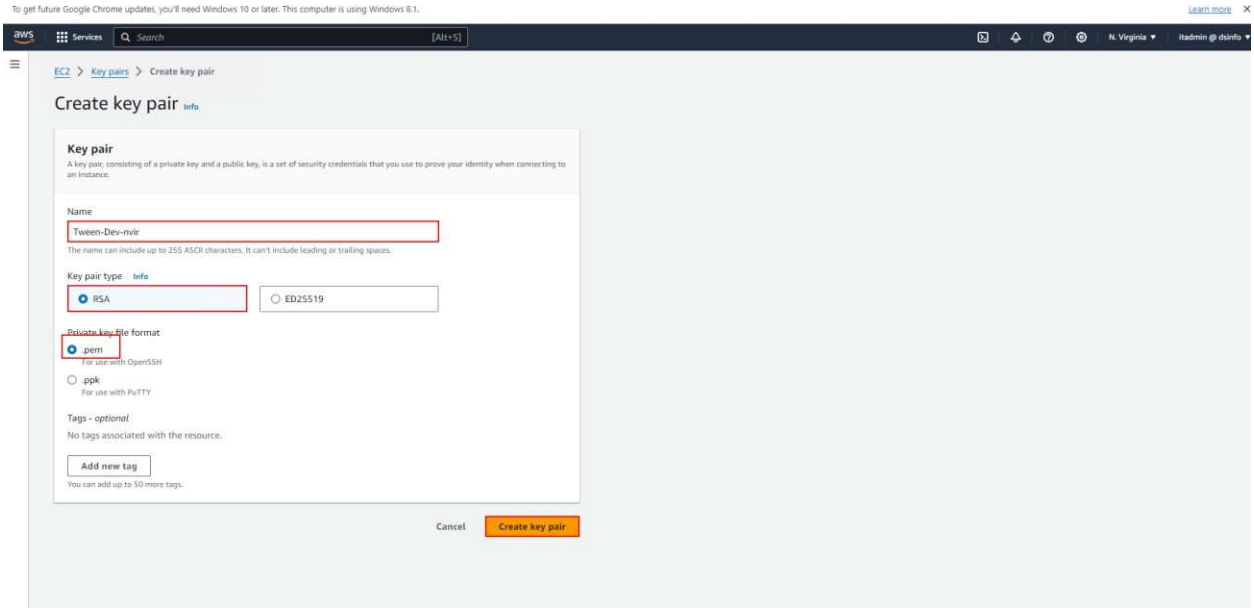
When you launch an instance, you can select and configure all the **requirements** such as the **Security Group**, **Key Pair**, and more.

How to Create a Key Pair

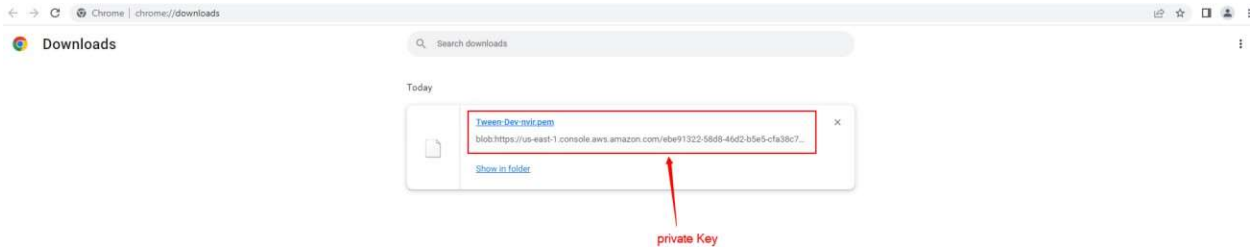
To create a **Key Pair**, go to the **Key Pairs** section and click on the **Create Key Pair** button.



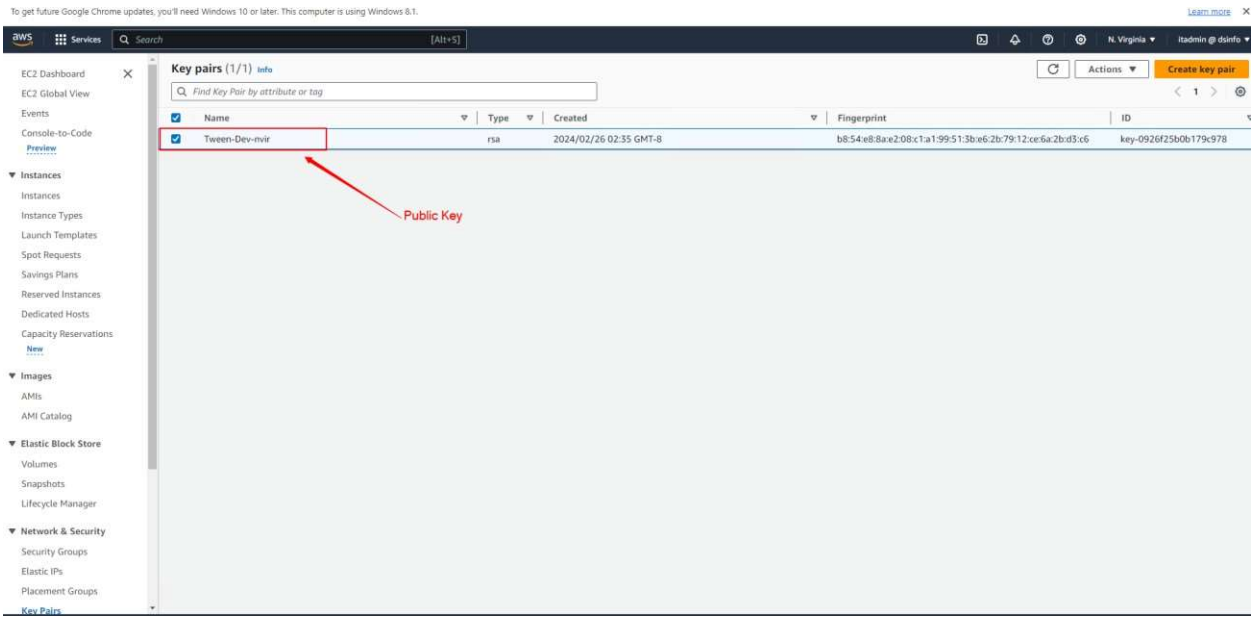
At this stage, specify a **Key Pair Name**, set the key type to **RSA**, and choose the **PEM** format for the private key. Finally, click on the **Create Key Pair** button.



Then, at this stage, download the **Private Key**.



The **Public Key** will be injected into your instance.



The **Private Key** must match your **Public Key**.

How to Create a Security Group

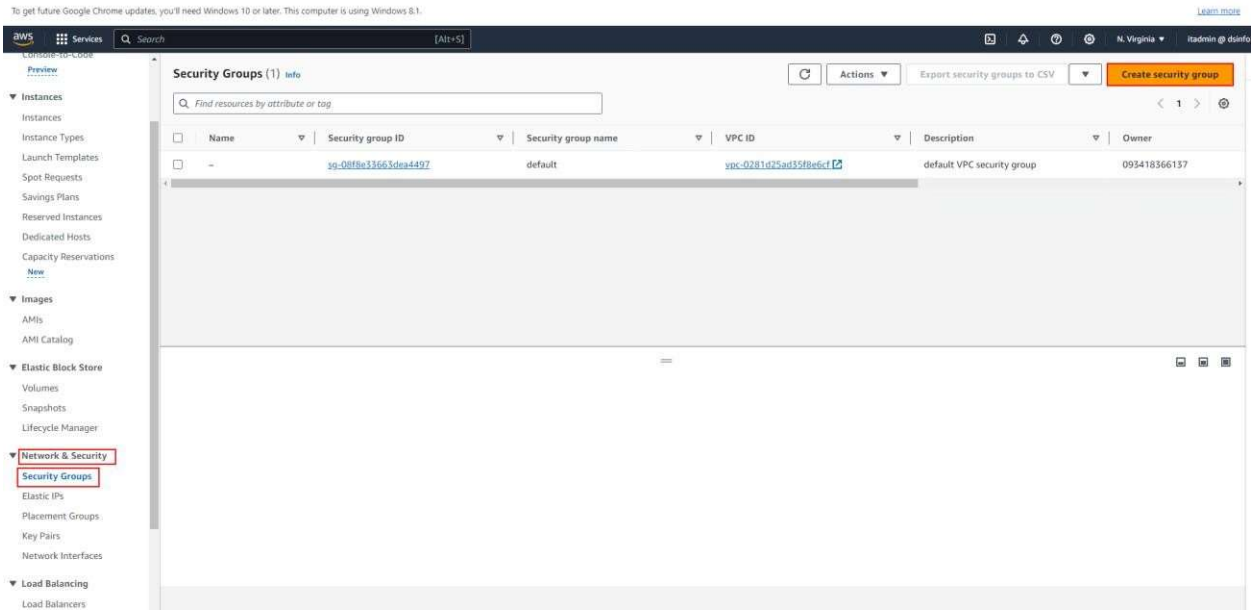
A **Security Group** is a **firewall**, and we can create it **before** launching an EC2 instance.

You can apply a single **Security Group** to multiple instances.

A **Security Group** works in a **stateful** manner. For example, if you allow access to port **22** for anyone in the **Inbound Rules**, the same rule is automatically applied to the **Outbound Rules**, so you don't need to define outbound rules separately.

To create a **Security Group**, follow these steps:

First, go to the **Security Groups** section and click on the **Create Security Group** button.



At this stage, you need to specify a **name** for the Security Group, and then define its **rules**. There are two types of rules in a Security Group:

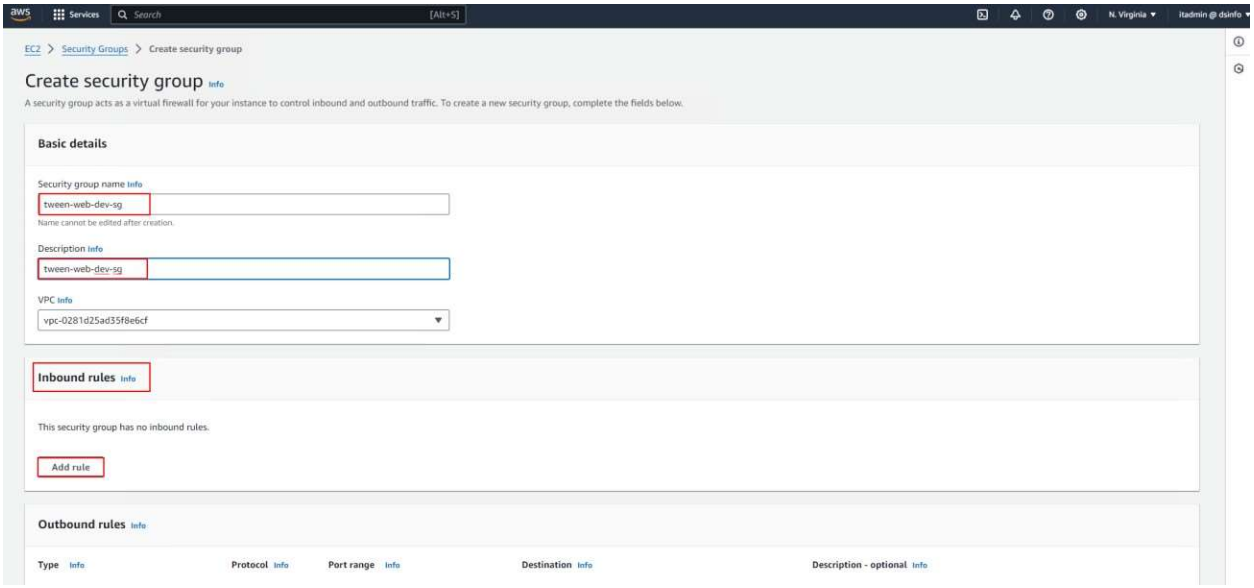
-Inbound Rule

This rule is for **inbound traffic** to the instance.

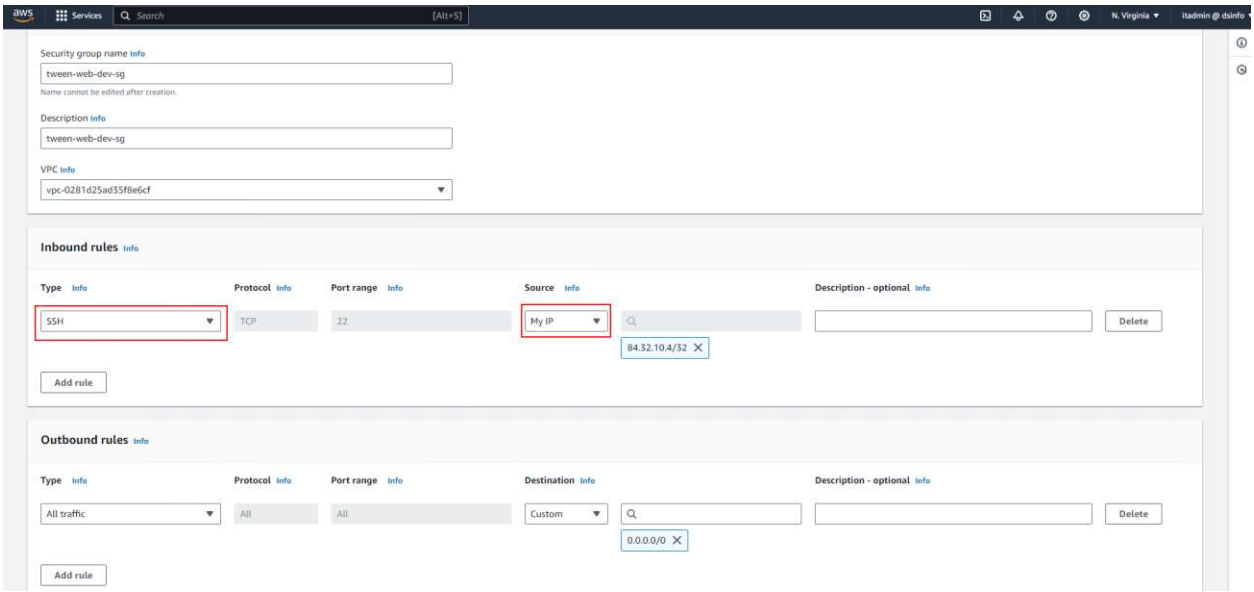
-Outbound Rule

This rule is for **outbound traffic** from the instance and usually doesn't need to be modified.

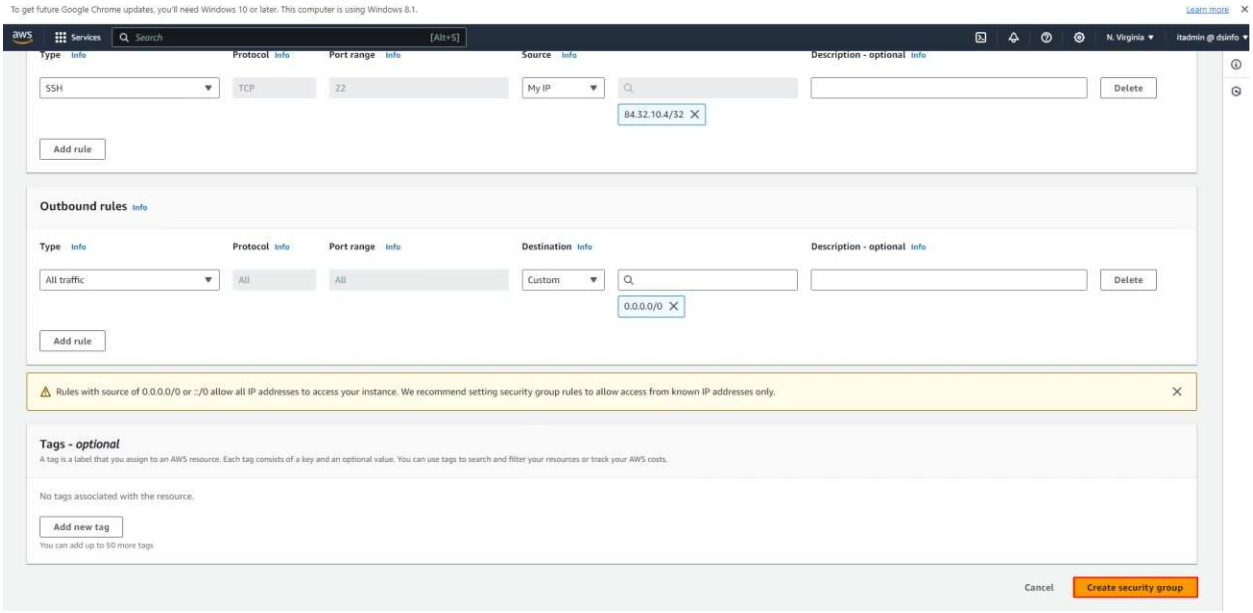
In this section, we intend to **open access to all inbound ports**. To do this, click on the **Add Rule** button under **Inbound Rules**, and then define the rule accordingly.



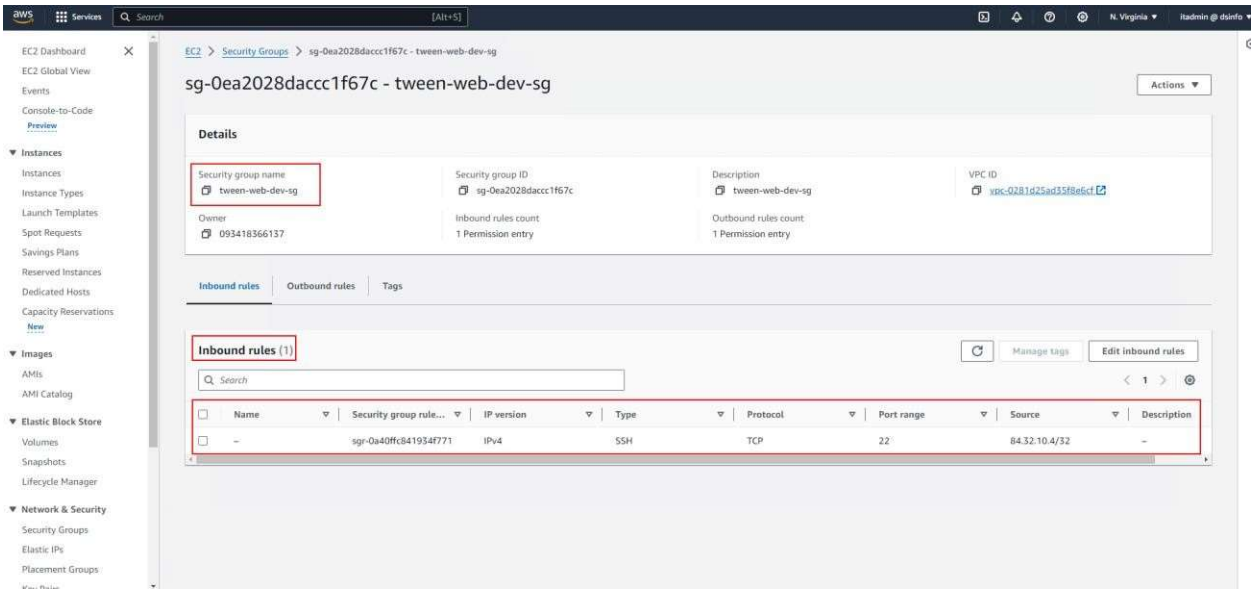
After clicking the **Add Rule** button, add the following rule to allow **SSH access** to the instance's IP



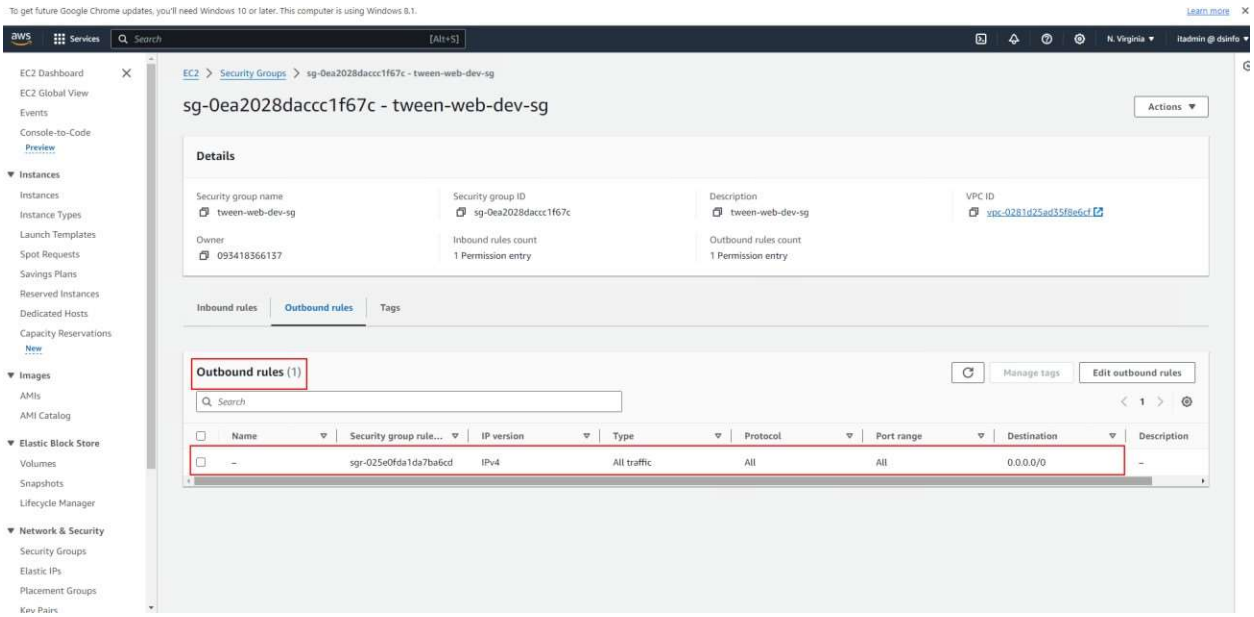
Finally, to create the **Security Group**, click on the **Create Security Group** button.



As shown in the image below, the **Security Group** has been successfully created.



As shown in the image below, the **Security Group** has been successfully created.

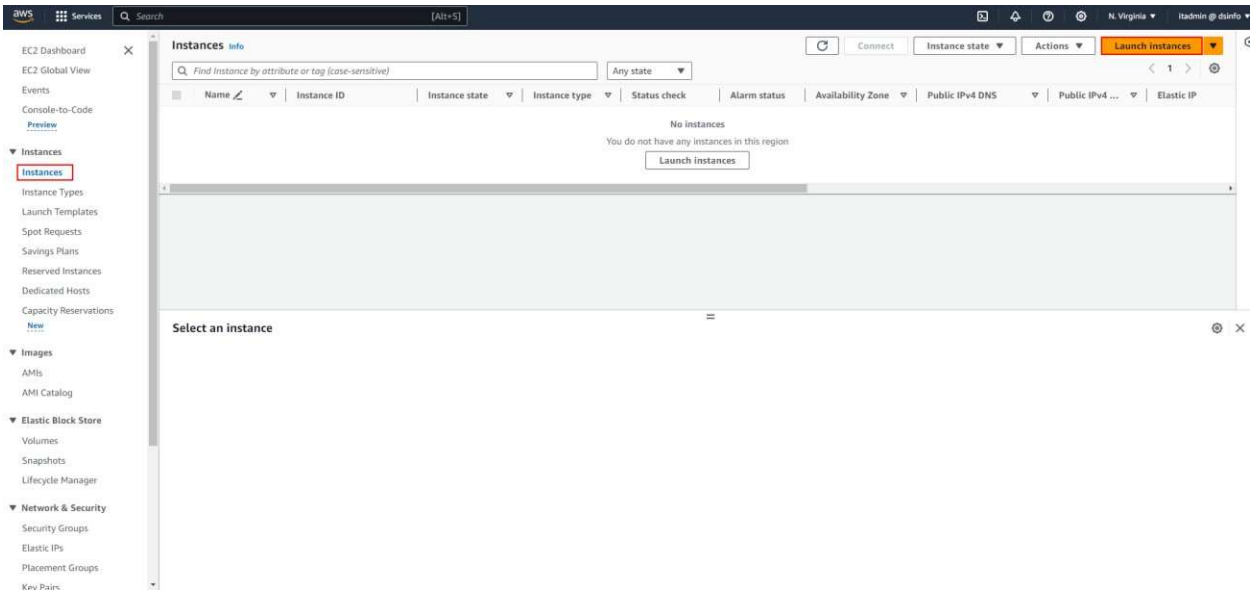


This **Outbound Rule** allows access to **any IP** and **any port** from the instance to the outside.

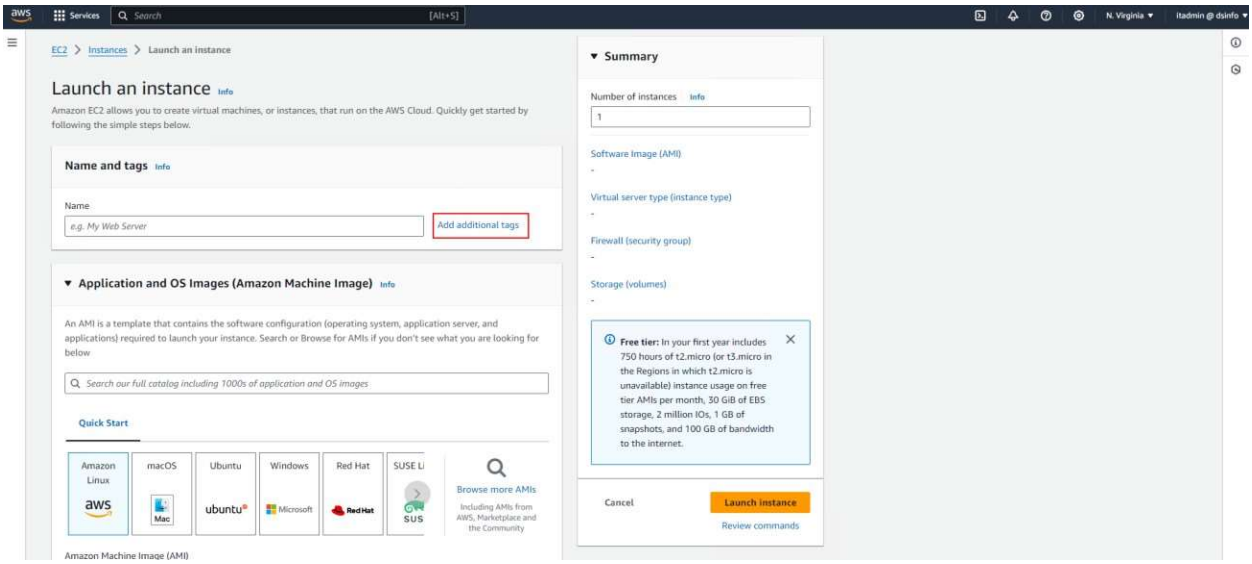
Deploying a Web Server on an EC2 Instance

Now, in this section, we intend to create an **EC2 instance** and use the **Key Pair** and **Security Group** that we previously created.

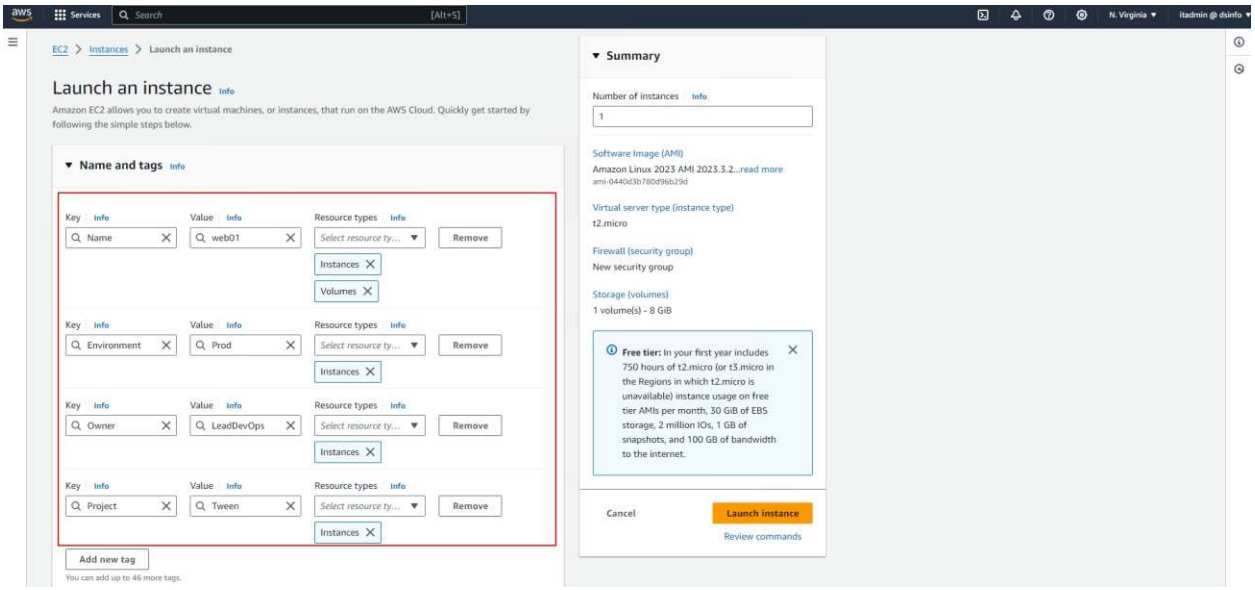
To create an EC2 instance, go to the **Instances** section and click the **Launch Instance** button.



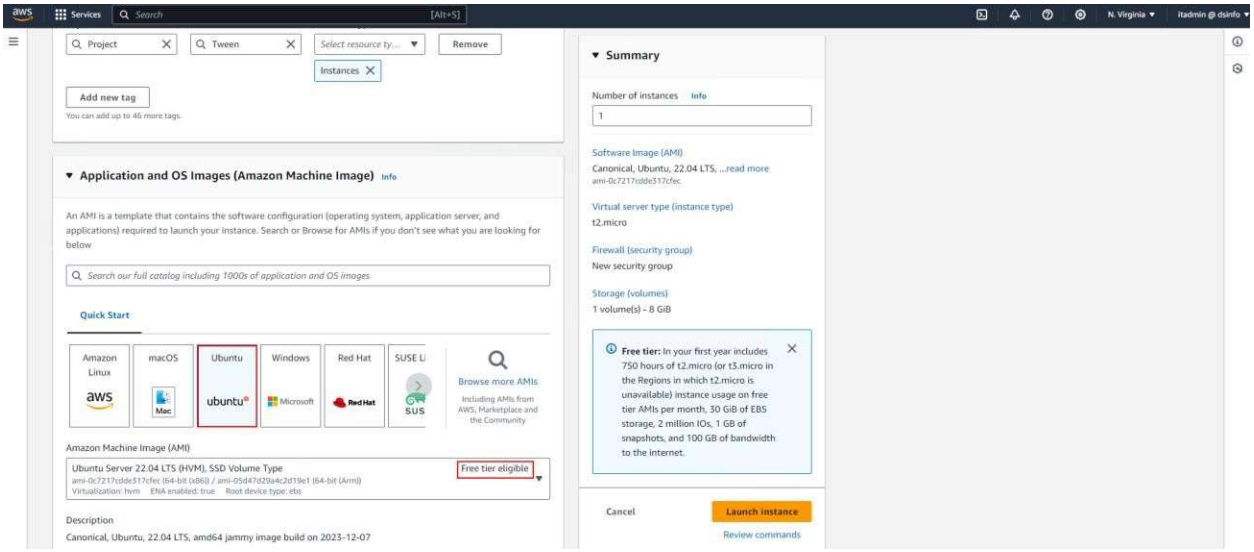
At this stage, click on the **Add Additional Tag** link to create a tag.



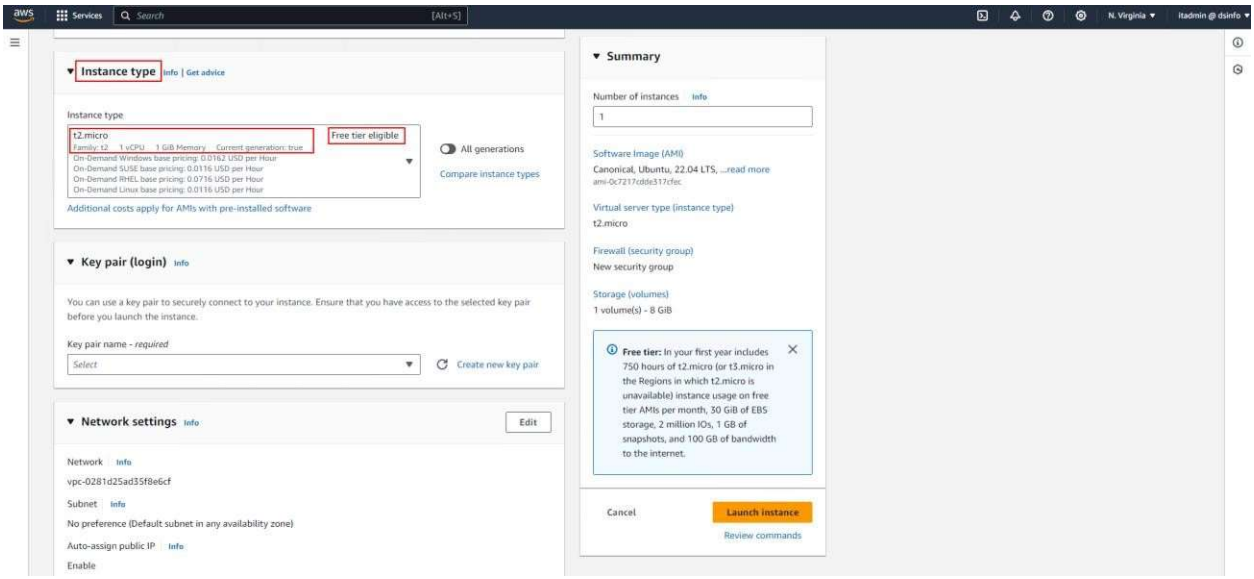
As shown in the image below, we have defined **4 tags** for the instance.



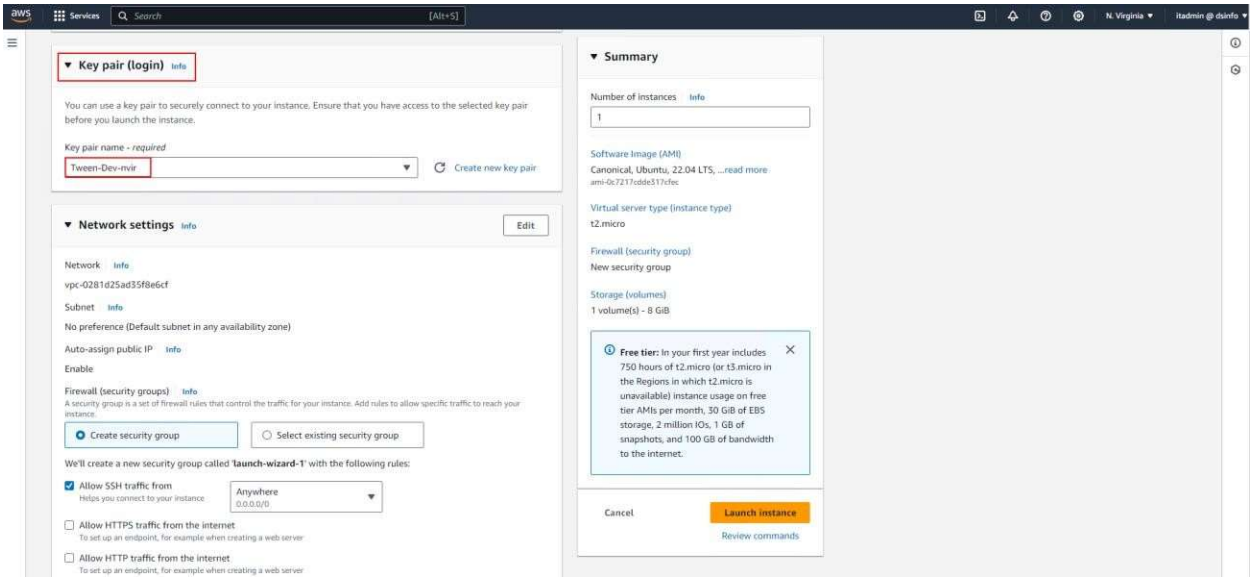
At this stage, you need to select the **AMI type**, and in this scenario, we are choosing **Ubuntu**, which, as you can see, is **Free Tier eligible**.



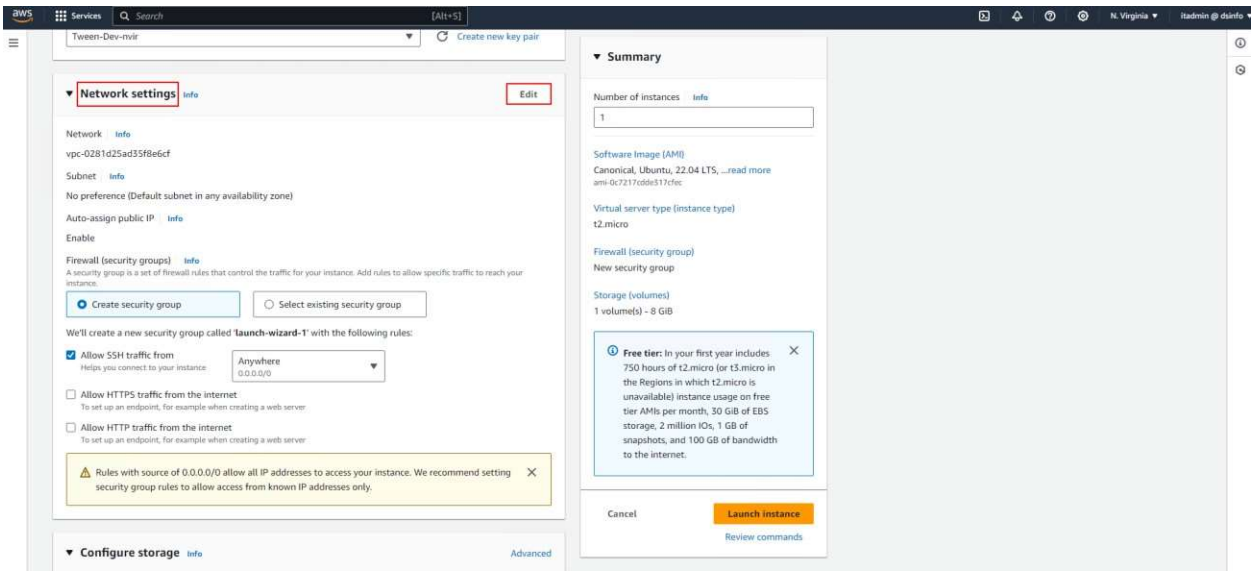
At this stage, you need to specify the **Instance Type**, which defines the hardware specifications of the instance. In this scenario, we are selecting **t2.micro**, which is **Free Tier eligible**.



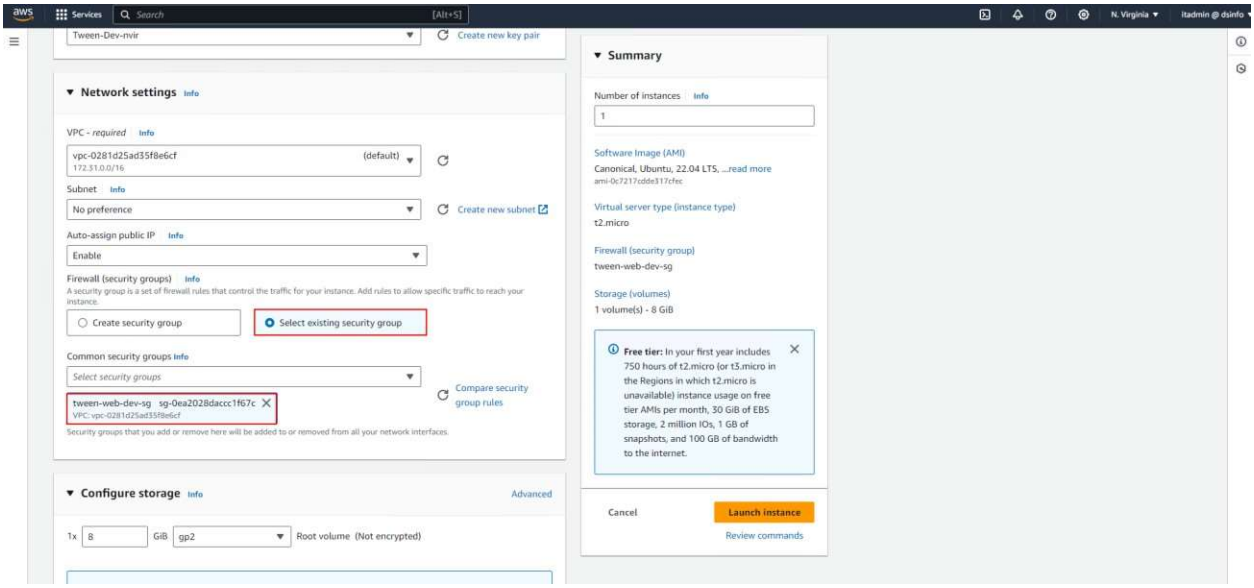
In this section, you need to specify the **Key Pair**, and we will apply the same Key Pair that we created earlier to this instance.



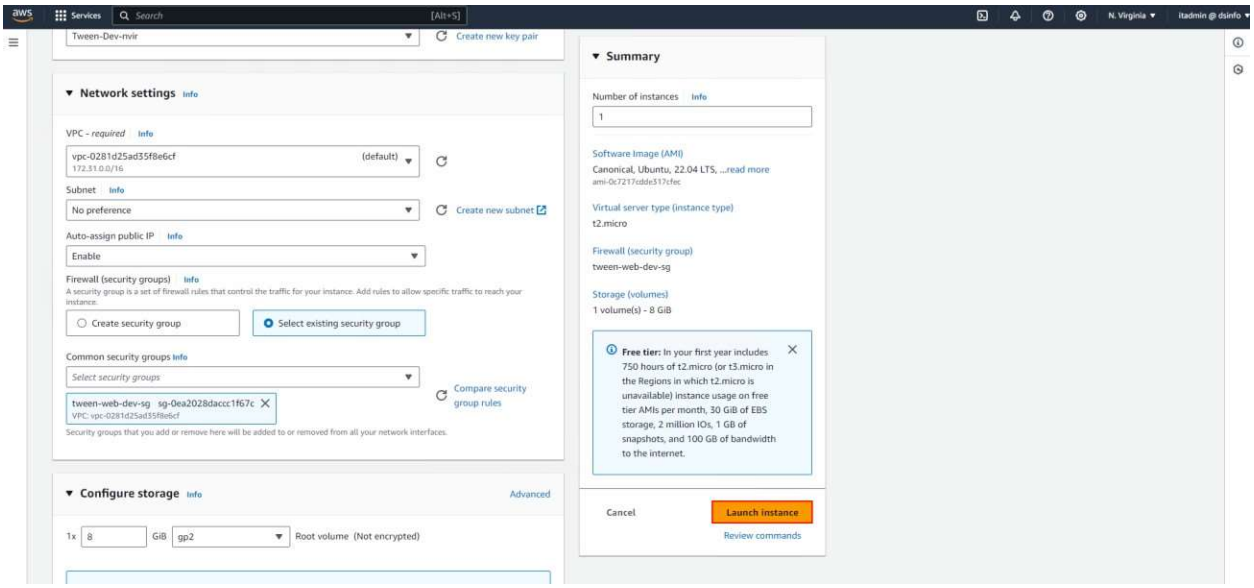
At this stage, to apply the **Security Group**, click on the **Edit** button in the **Network Settings** section.



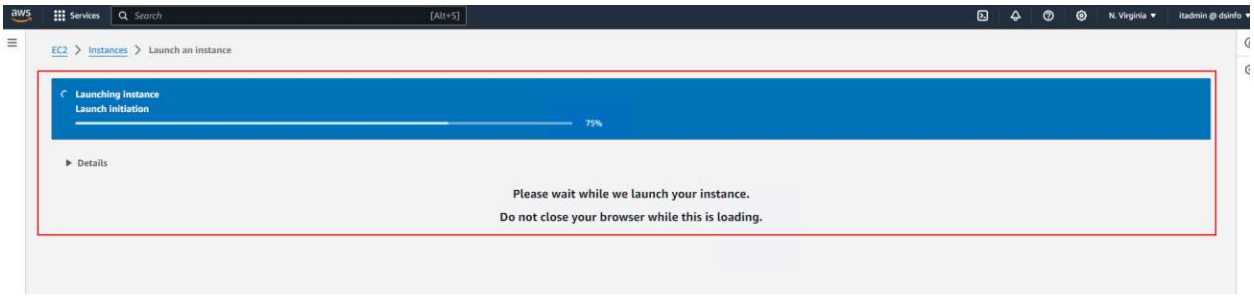
Then, select the **Select Existing Security Group** option, and choose the **Security Group** that we created in the previous steps.



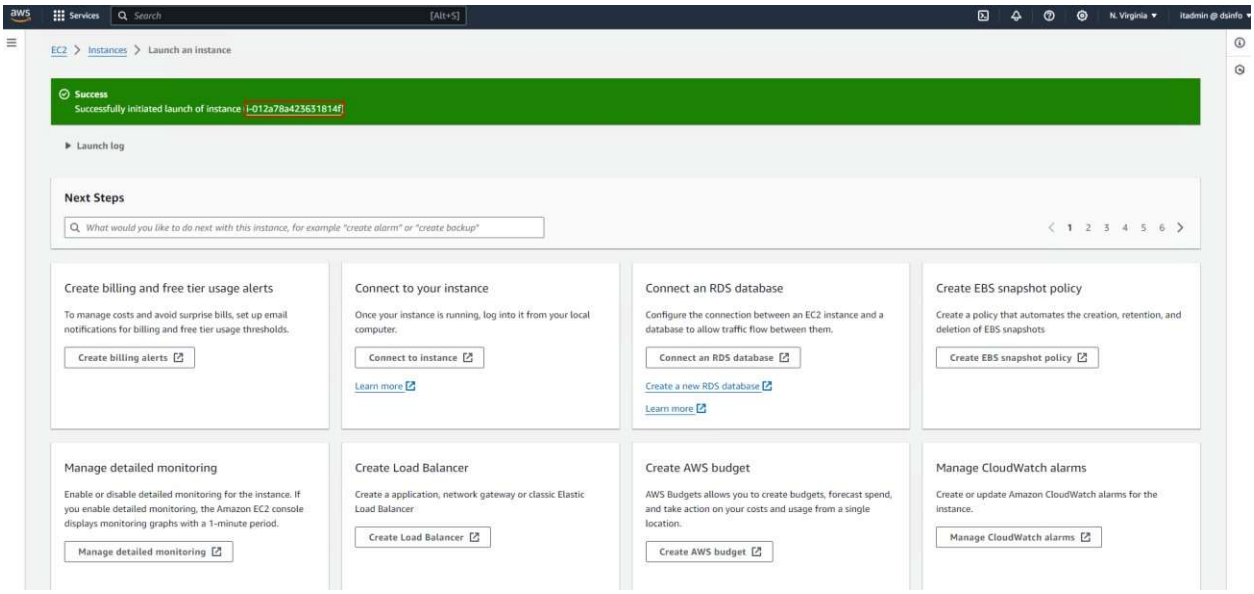
Finally, click on the **Launch Instance** button to create the instance.



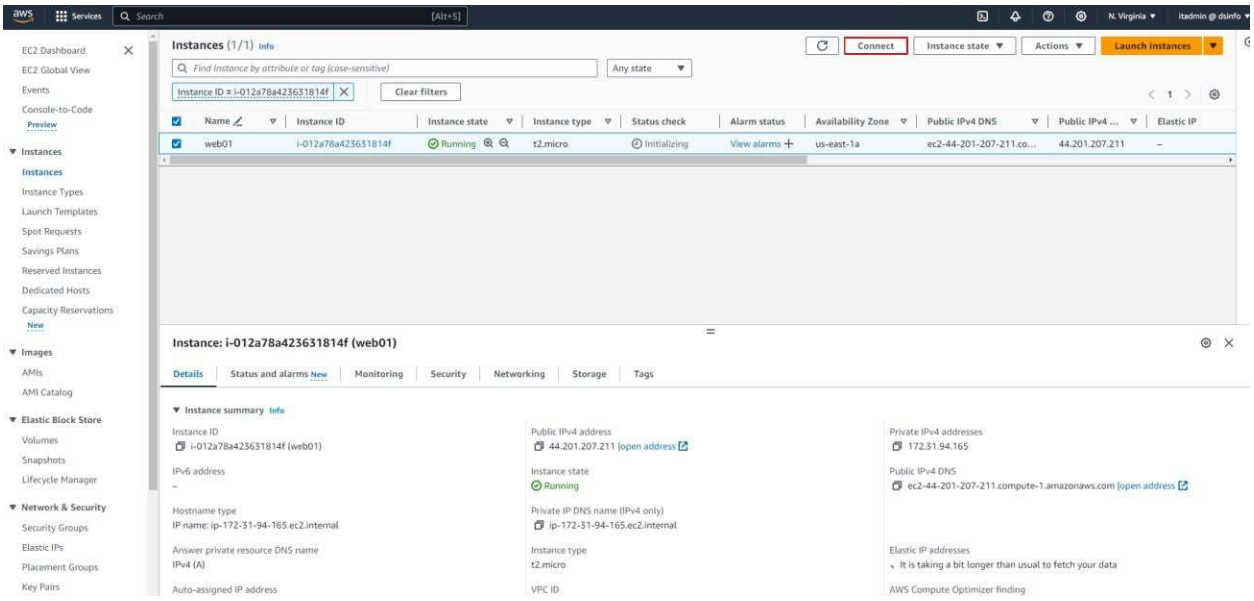
As shown in the image below, the instance is being created.



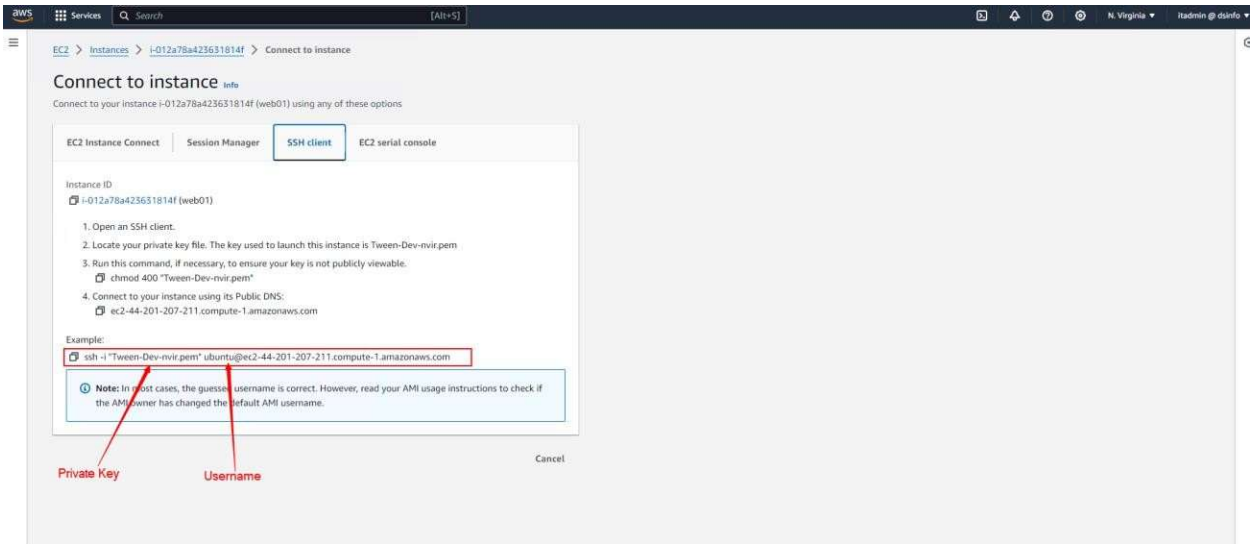
The instance has been created. To view the details and status of this instance, simply click on its **Instance ID**.



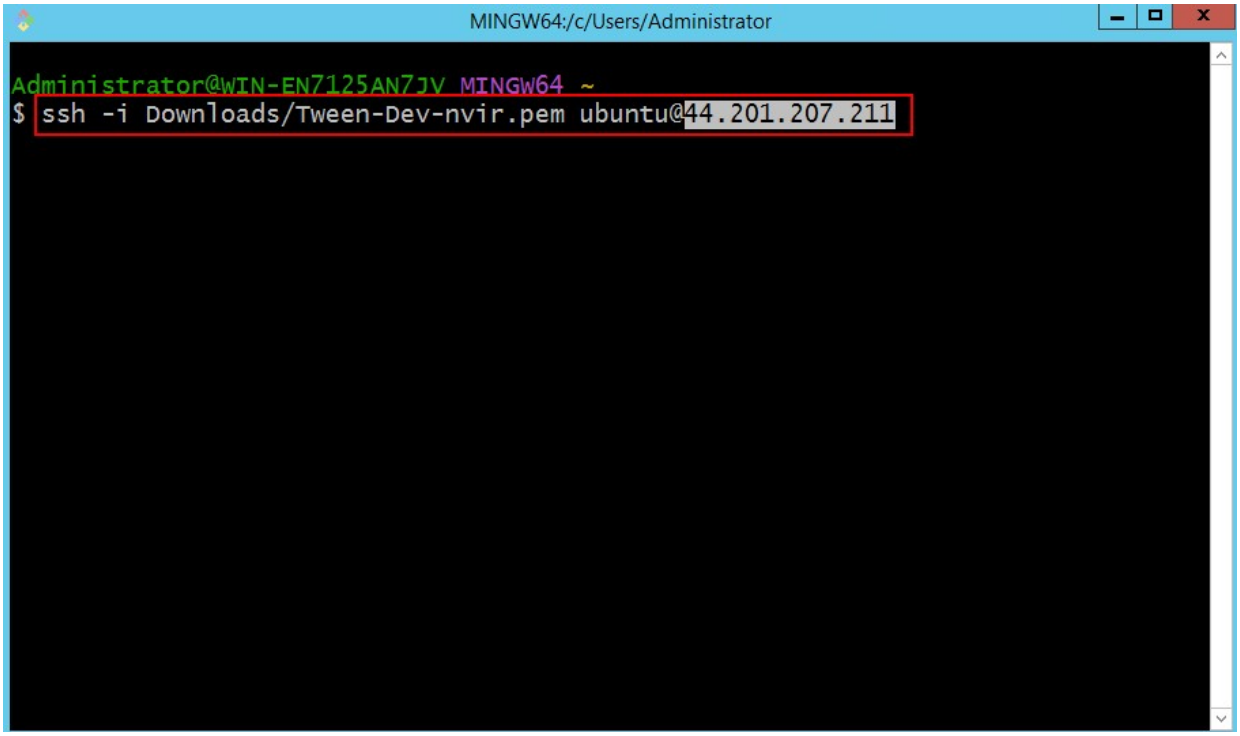
To connect to the instance, click on the **Connect** button.



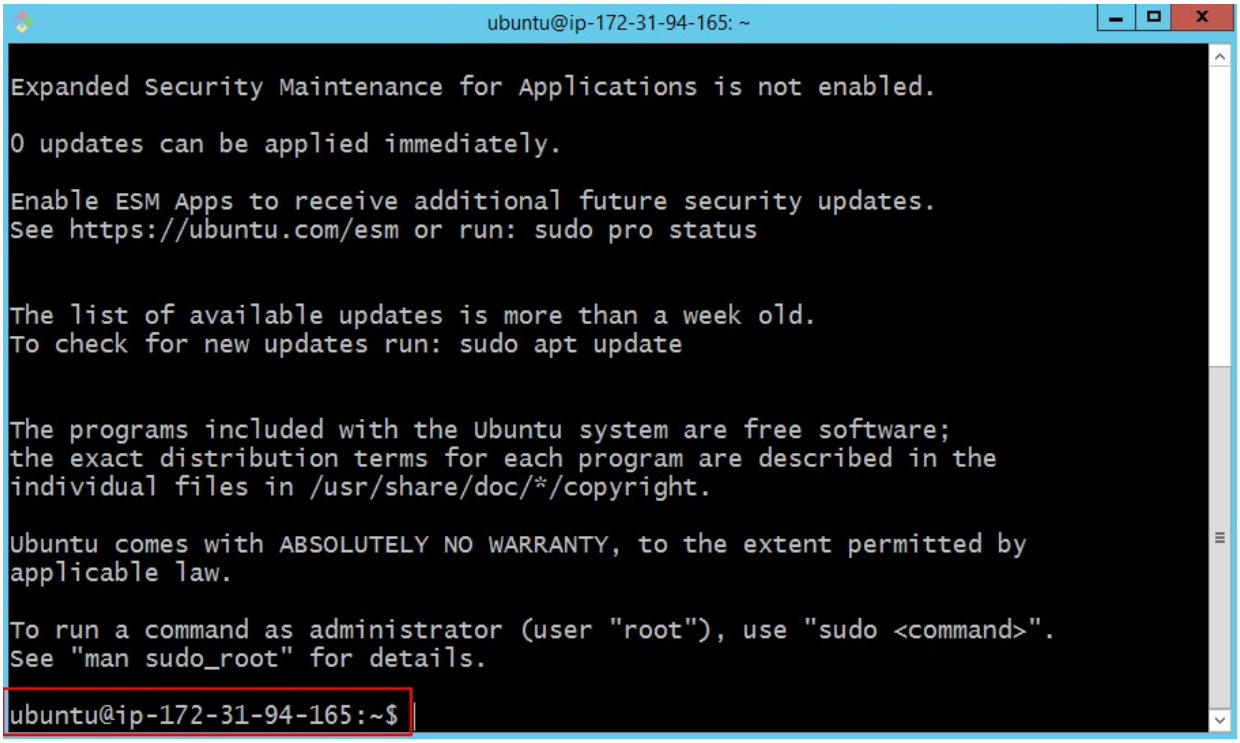
In the image below, you can see the required information for **SSH** to connect to the instance.



To connect to the instance from your system, you can use **SSH**. Make sure to use the **Private Key** associated with the instance for the connection.

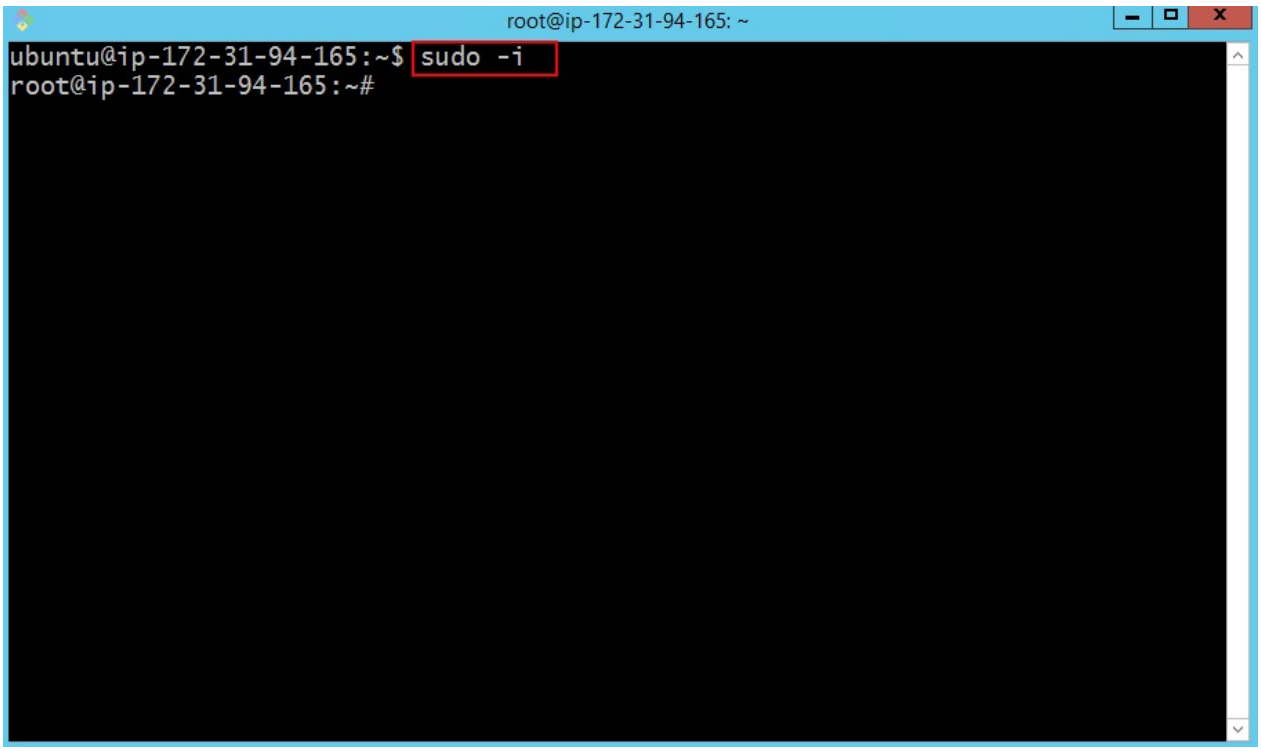


As shown in the image below, we have successfully connected to the **Ubuntu terminal** on the EC2 instance via **SSH**.

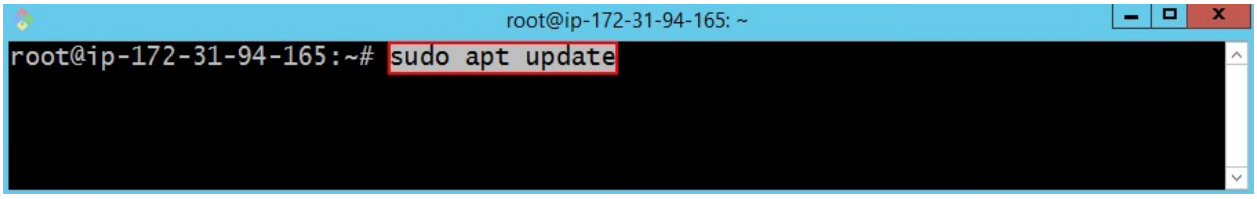


Now, in this section, we will install the **Tween website**, which is one of the templates from **Tooplate**, on this EC2 instance.

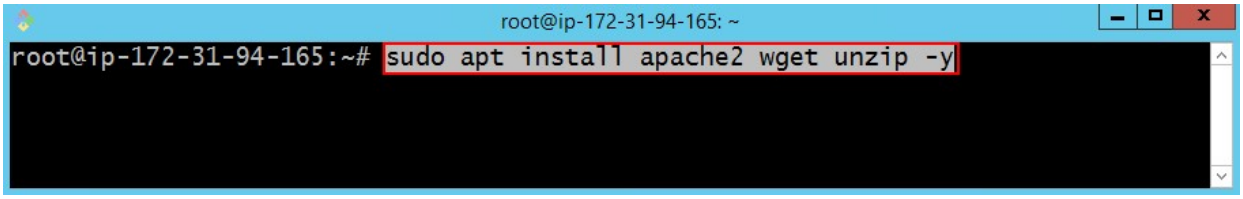
At this stage, use the following command to enter the **Root User** environment



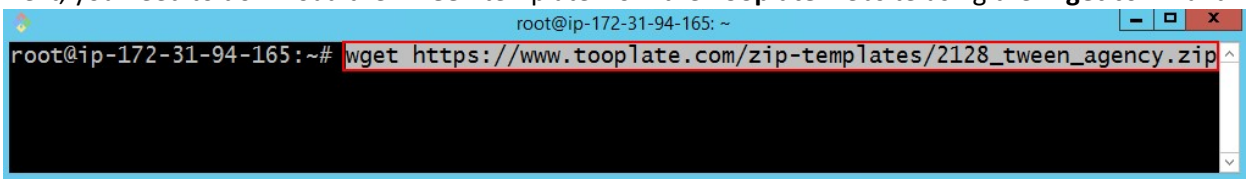
Then, use the following command to **update the repository** for the Linux instance



At this stage, use the following command to install the **Apache**, **Wget**, and **Unzip** packages

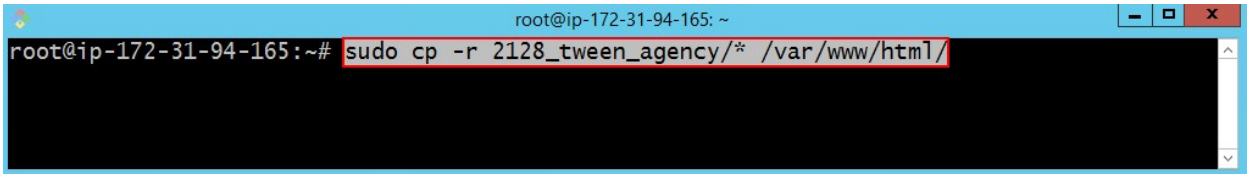


Next, you need to download the **Tween** template from the **Tooplate** website using the **Wget** command.

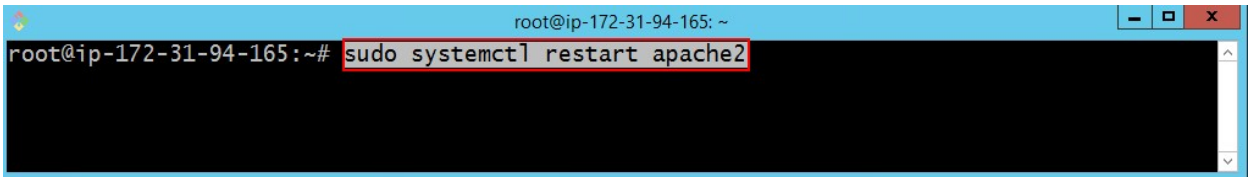


After downloading the **Tween** file, you need to unzip it using the following command

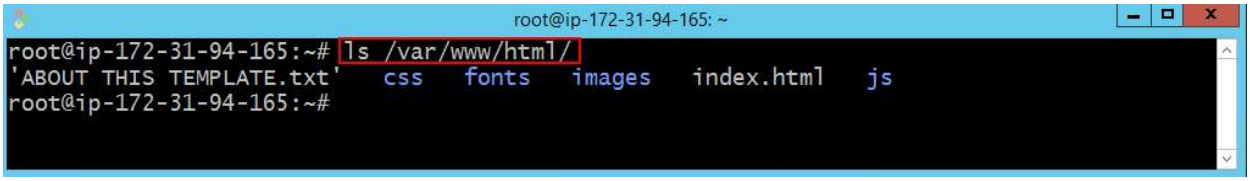
At this stage, you need to copy the unzipped contents of the **Tween** website to the web server's directory.



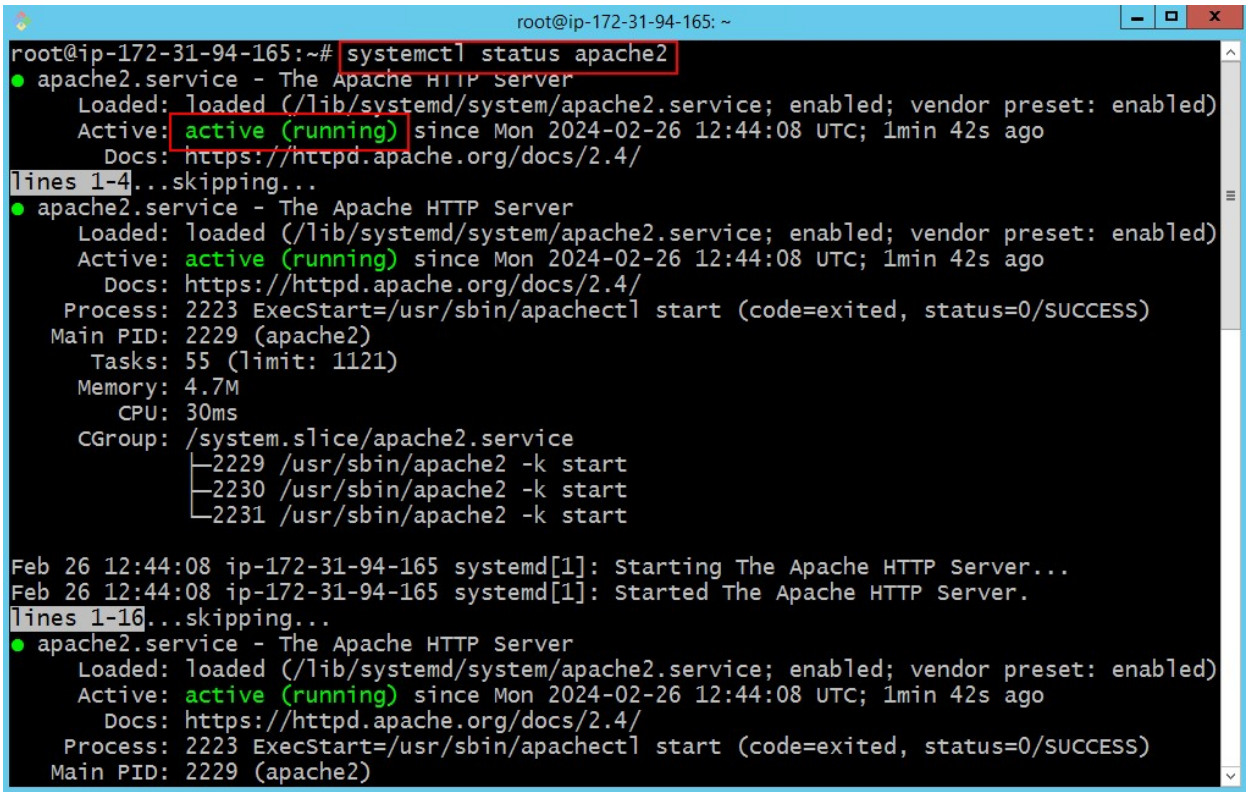
Finally, you need to restart the **Apache** service using the following command



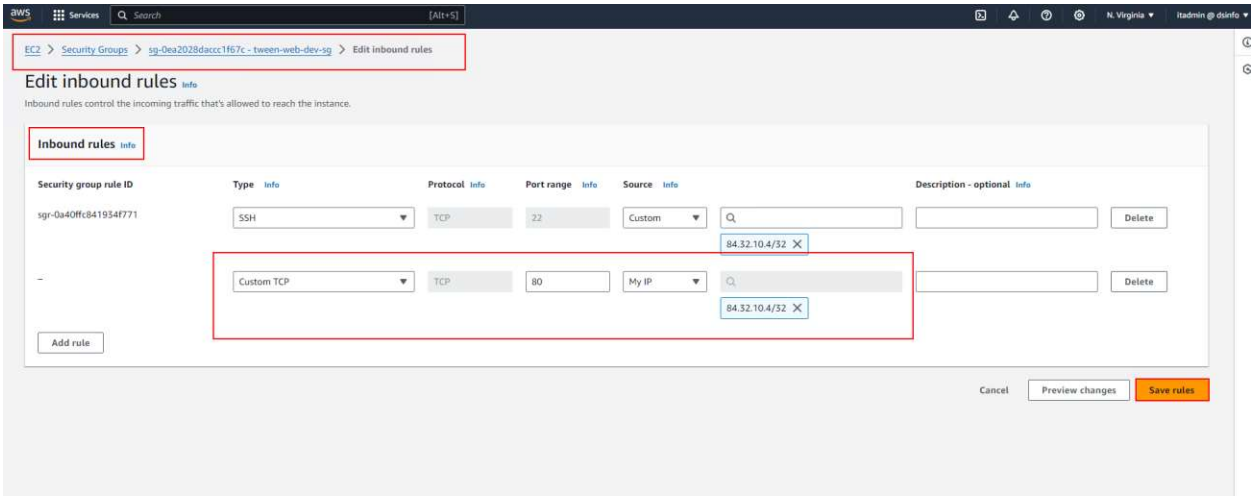
You can check if the **Tween** website files have been copied to the web server directory using the following command



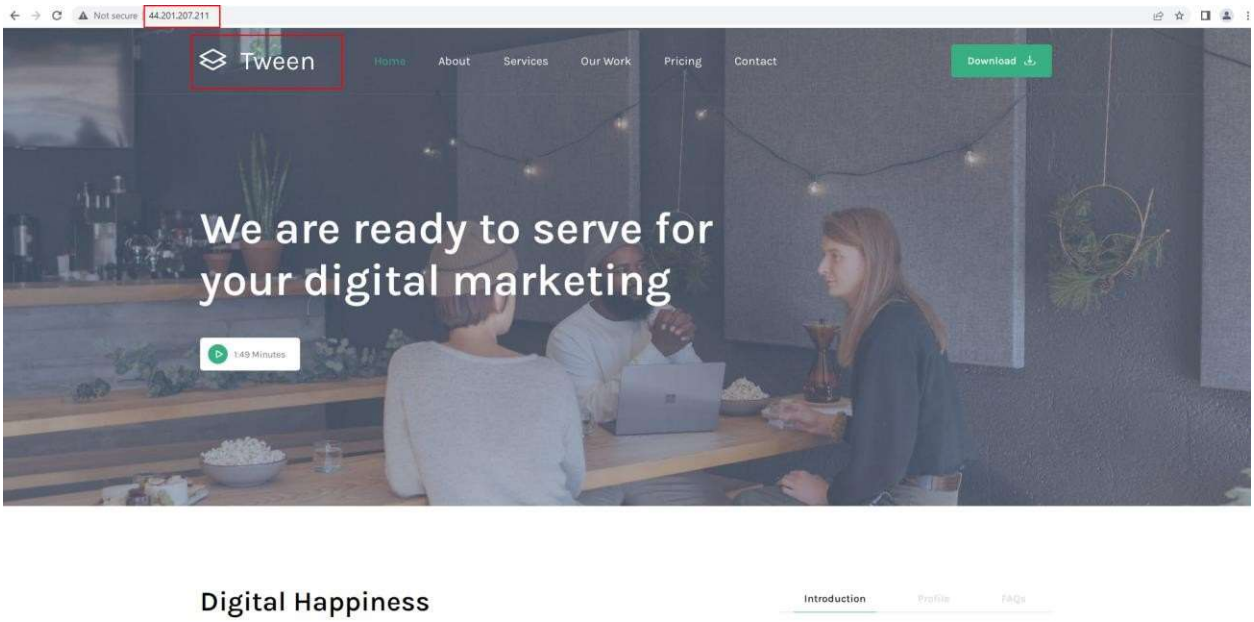
You can check if the **Apache** service is running by using the following command



At this stage, you need to open port 80 in the **Inbound Rule** to allow access to the website from the internet.



Now, we can access the **Tween** website by entering the **Public IP** of the instance in our browser.



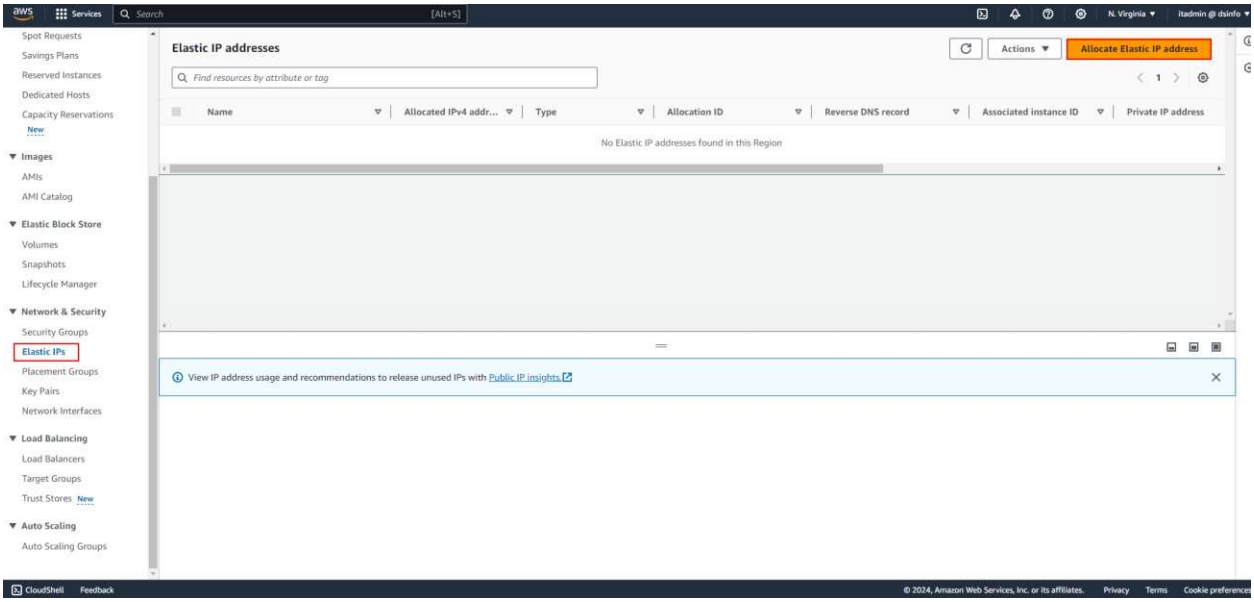
Introduction to Elastic IP

When you launch an EC2 instance, AWS automatically assigns a **Public IP**. However, when you stop the instance, the Public IP is released, and when the instance is started again, it is assigned a new **Public IP**.

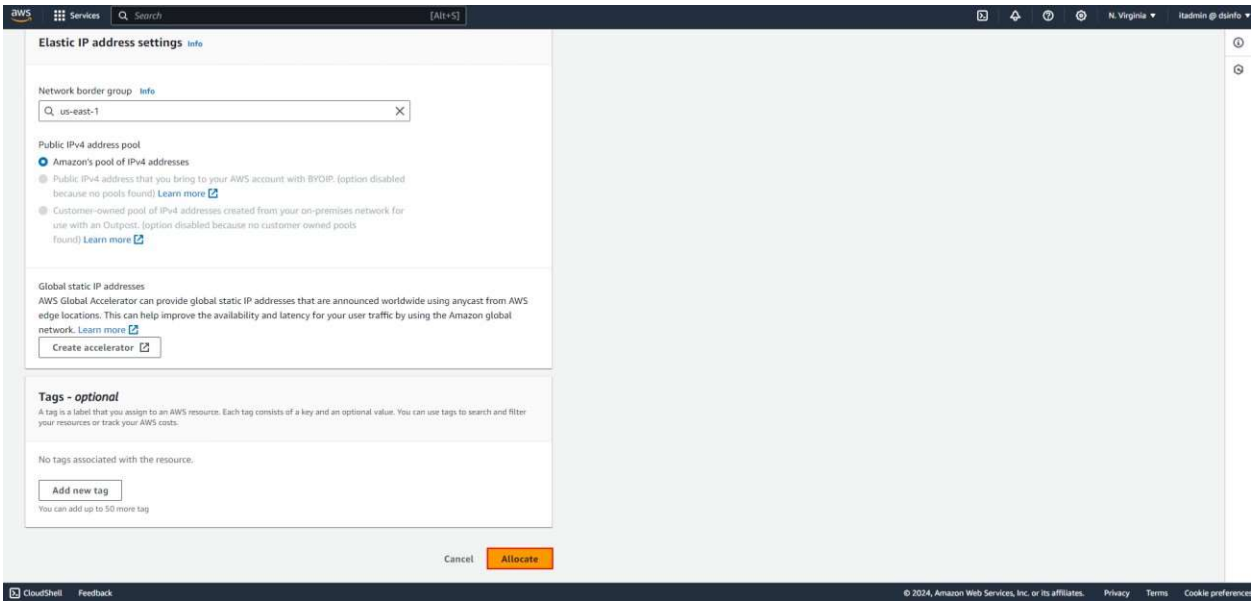
To use a **static Public IP**, you need to use AWS's **Elastic IP** feature, which provides you with up to **5 Public IPs**. If you need more IPs, you must submit a support ticket to request additional IPs.

How to Create an Elastic IP

To create an **Elastic IP**, simply click on the **Elastic IPs** link, then click the **Allocate Elastic IP Address** button.

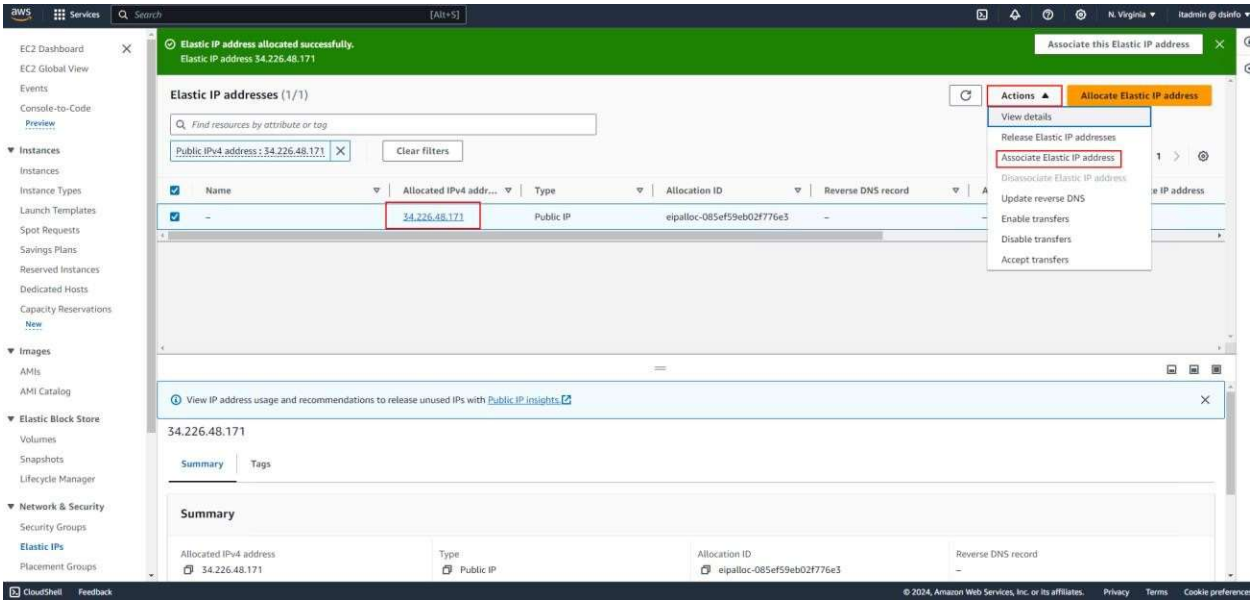


Finally, click on the **Allocate** button.

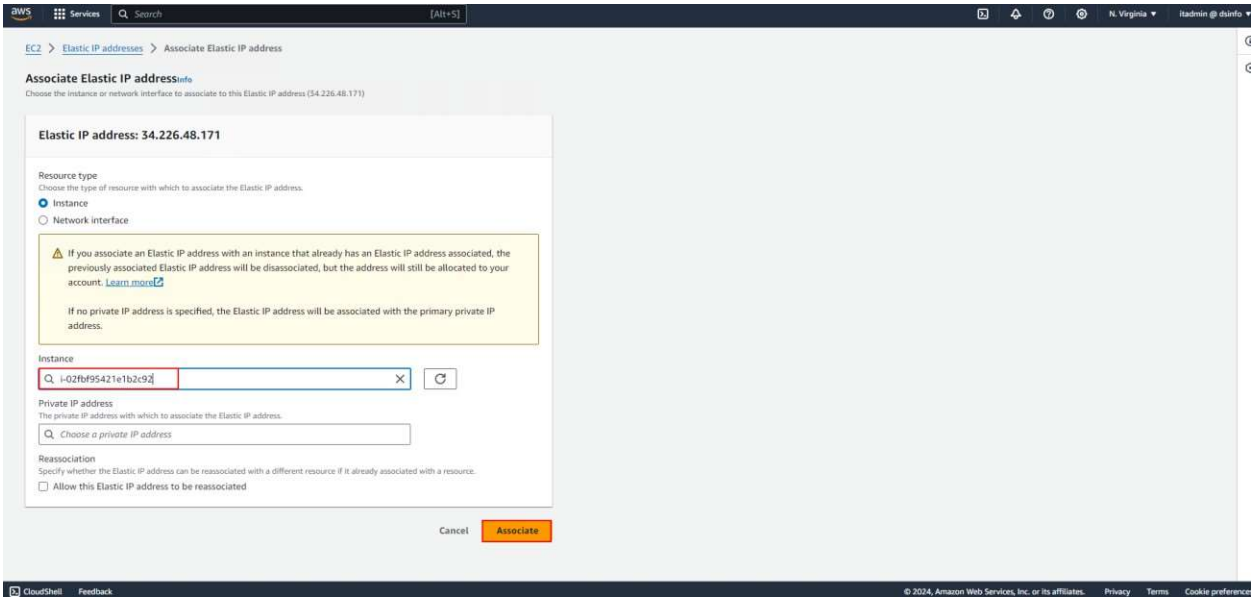


By creating an **Elastic IP**, this **Public IP** will be reserved for us.

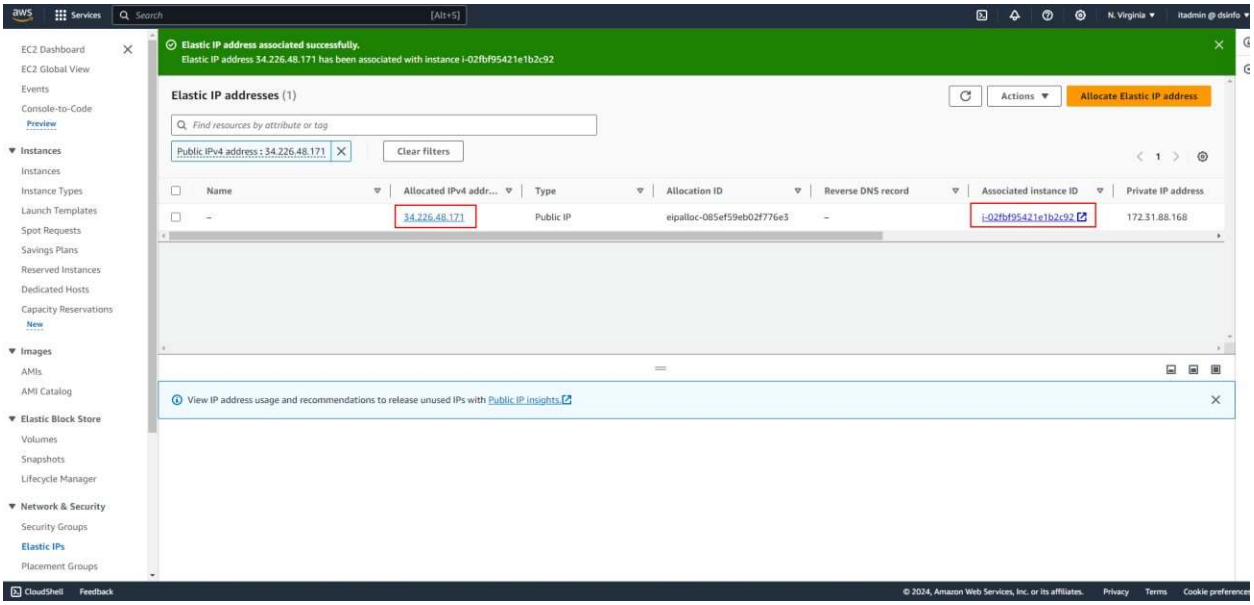
To associate the **Elastic IP** with an instance, go to the **Actions** menu and click on **Associate Elastic IP Address**.



Next, in the **Instance** section, select the desired instance, and then click the **Associate** button.



As shown in the image below, the **Instance ID** has been successfully associated with our **Elastic IP**.



Note:

All network settings, such as **Security Group**, **Elastic IP**, etc., are applied to the **Network Interface** on the instance.

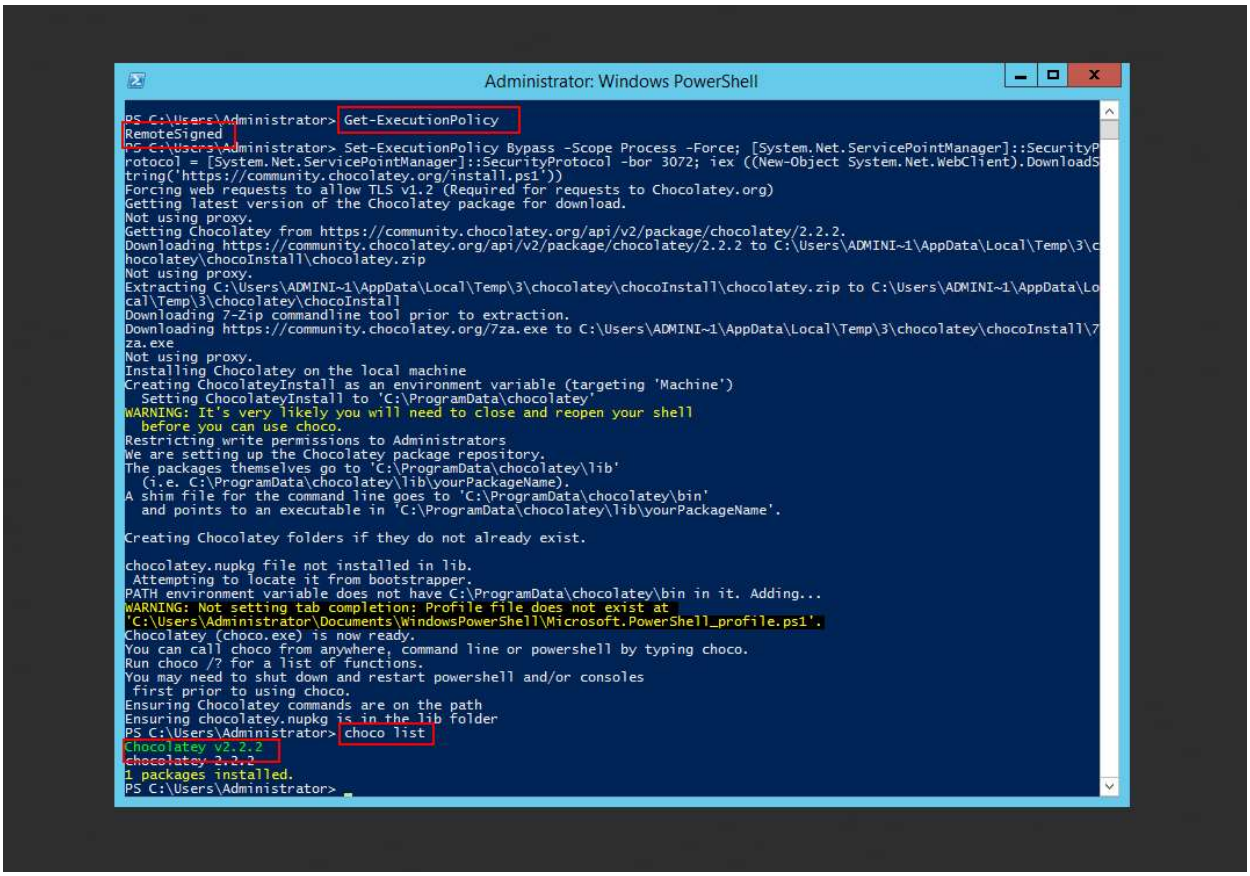
Introduction to AWS CLI

We can perform all the tasks that we do through the **console** or **graphical interface** using the **CLI** (Command Line Interface). As a DevOps professional, you should be familiar with these commands and work with them efficiently.

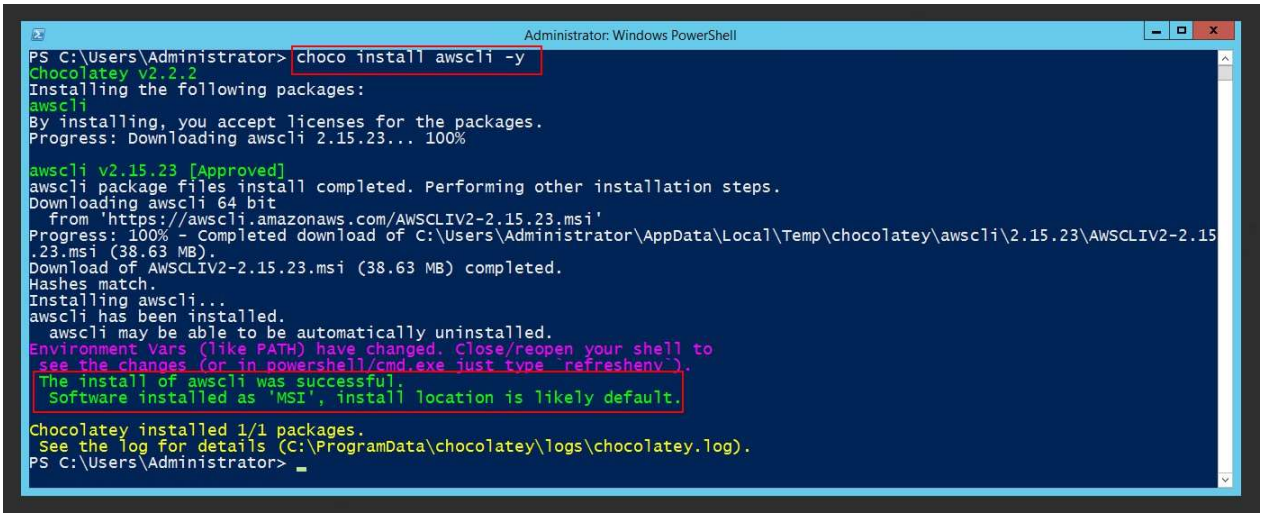
Installing AWS CLI

To install AWS CLI, you first need to install the **Chocolatey** tool on your **Windows PowerShell** environment, as shown in the image below.

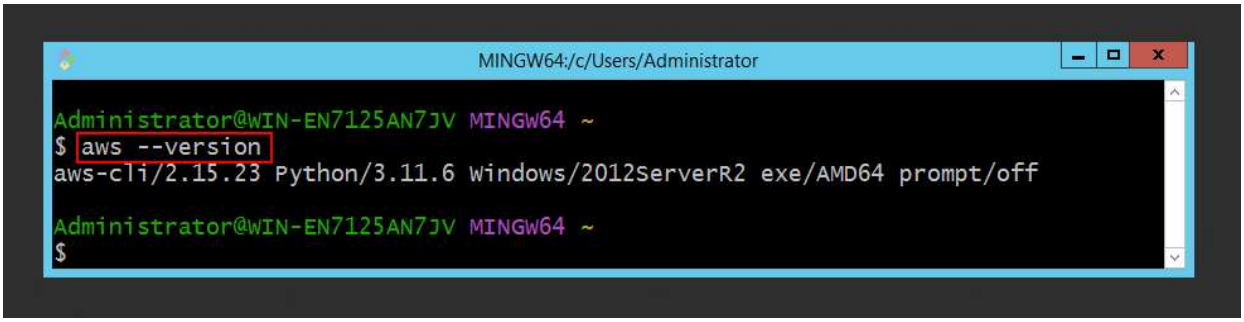
Set-ExecutionPolicy Bypass -Scope Process -Force;
[System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))



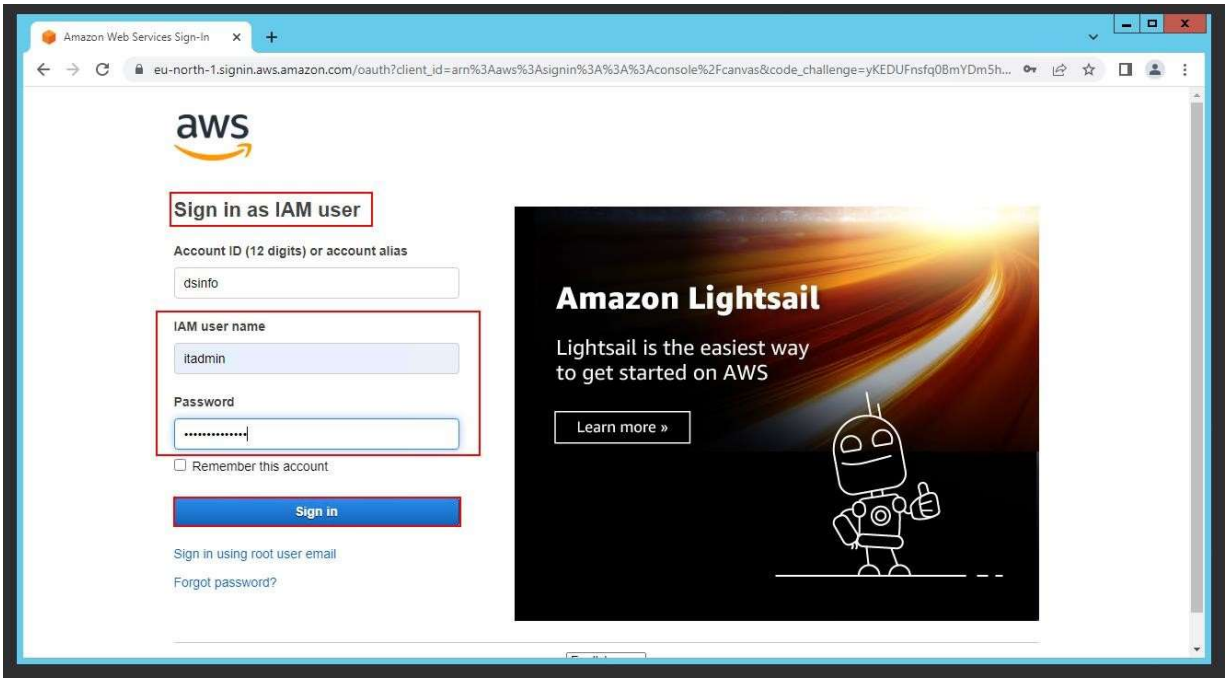
After installing **Chocolatey** in the **PowerShell** environment, you need to install the **AWS CLI** tool using the following command



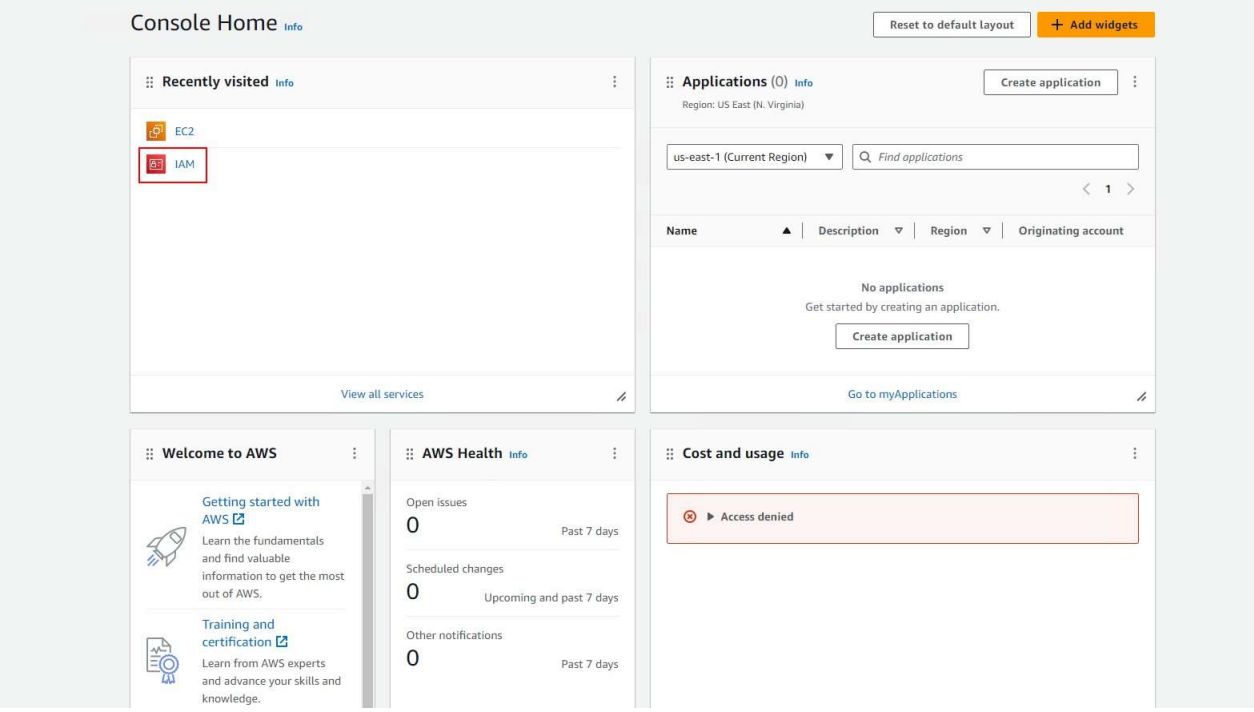
To check if **AWS CLI** has been installed correctly, use the following command in the **Git Bash** environment



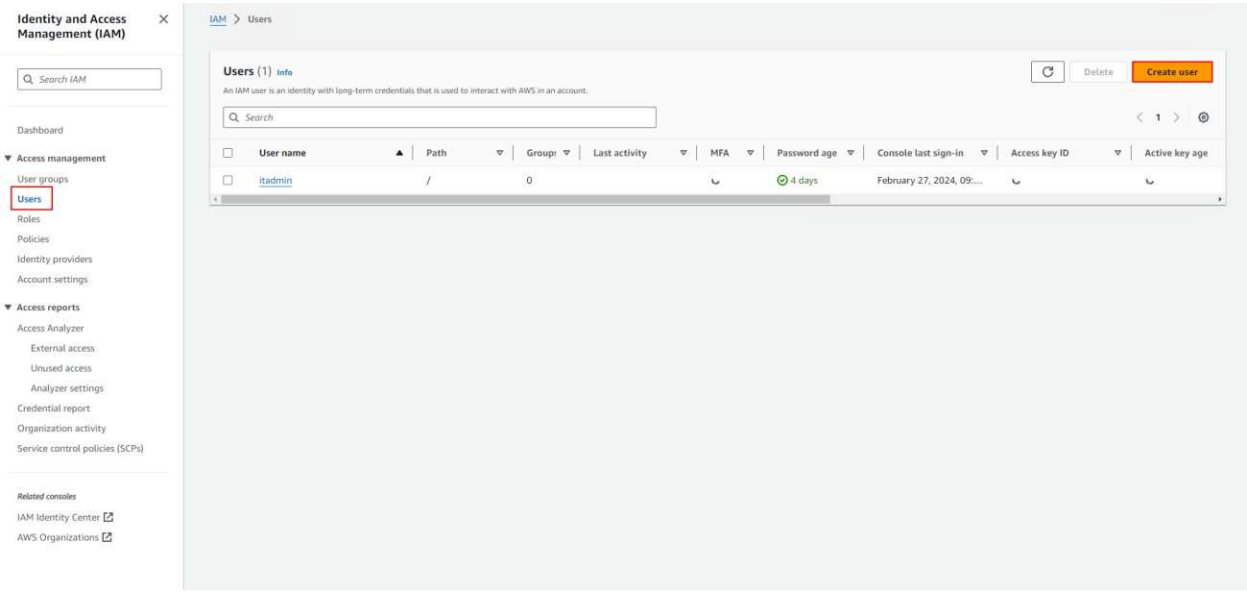
At this stage, you need to log in to the **AWS Management Console** using an **IAM User** to create a user for accessing the AWS CLI environment.



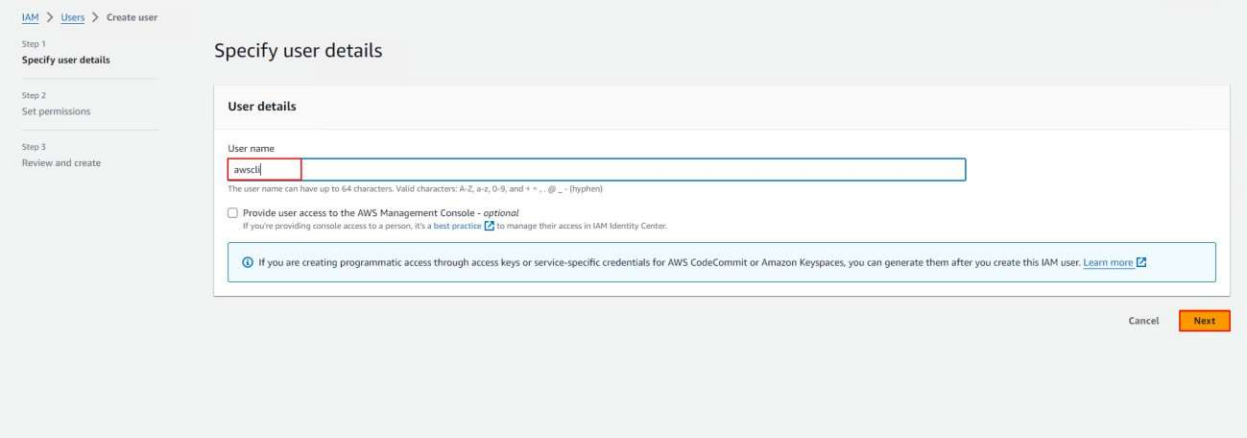
At this stage, click on **IAM** to enter the **IAM Console** page.



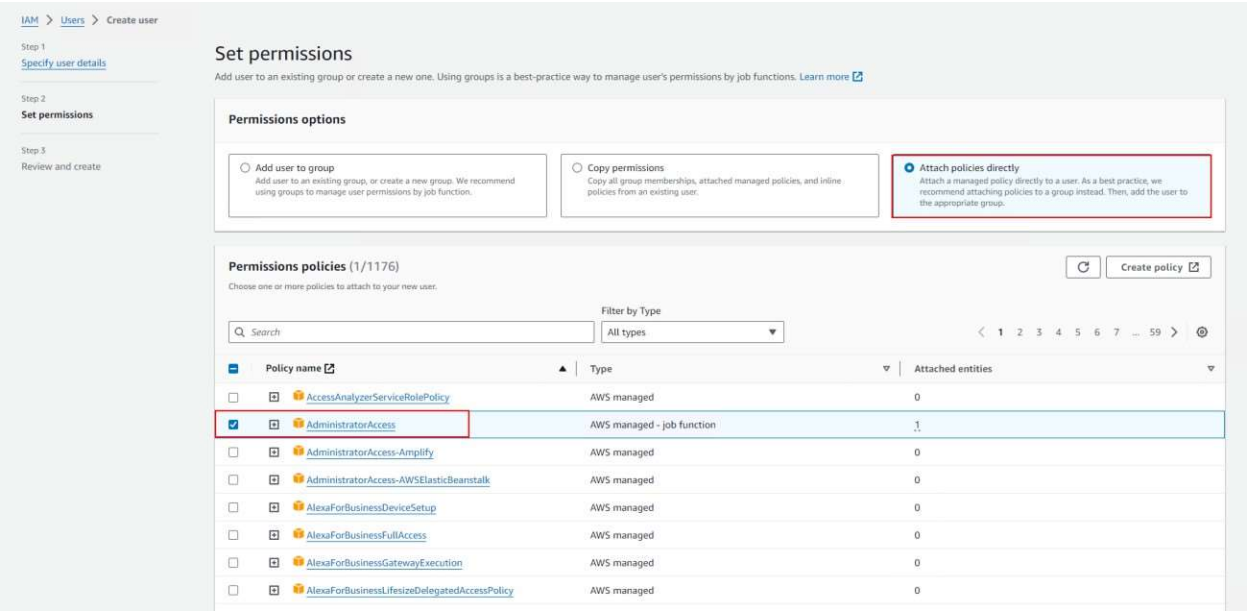
To create a user for AWS CLI, click on the **Users** link, and then click on the **Create User** button.



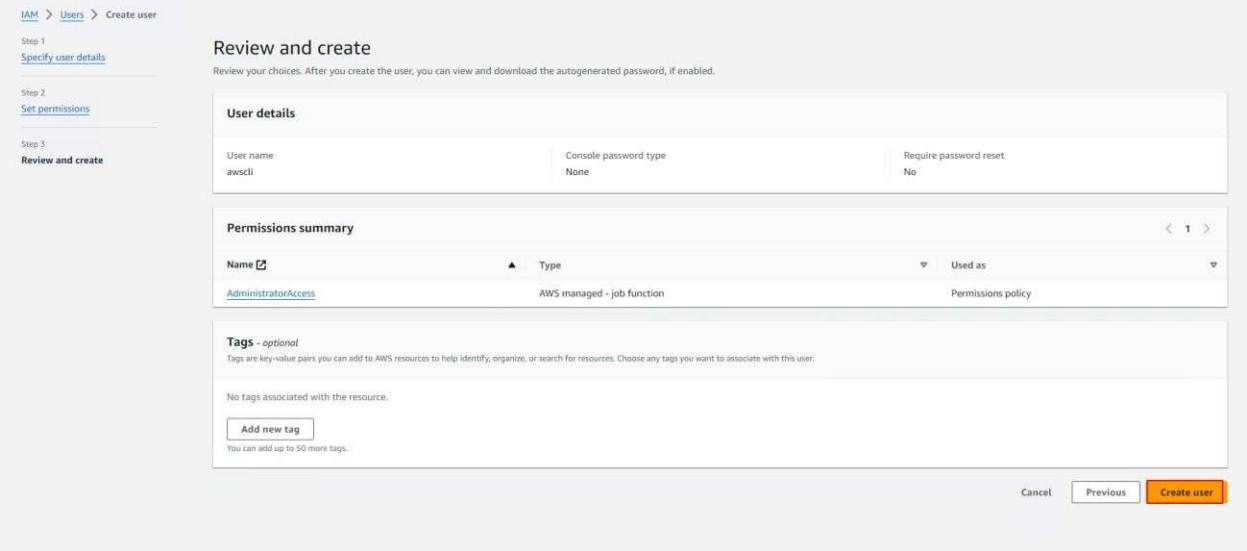
In the **Username** section, define a user but make sure the option **User Access to AWS Management Console** is not selected. Then, click the **Next** button.



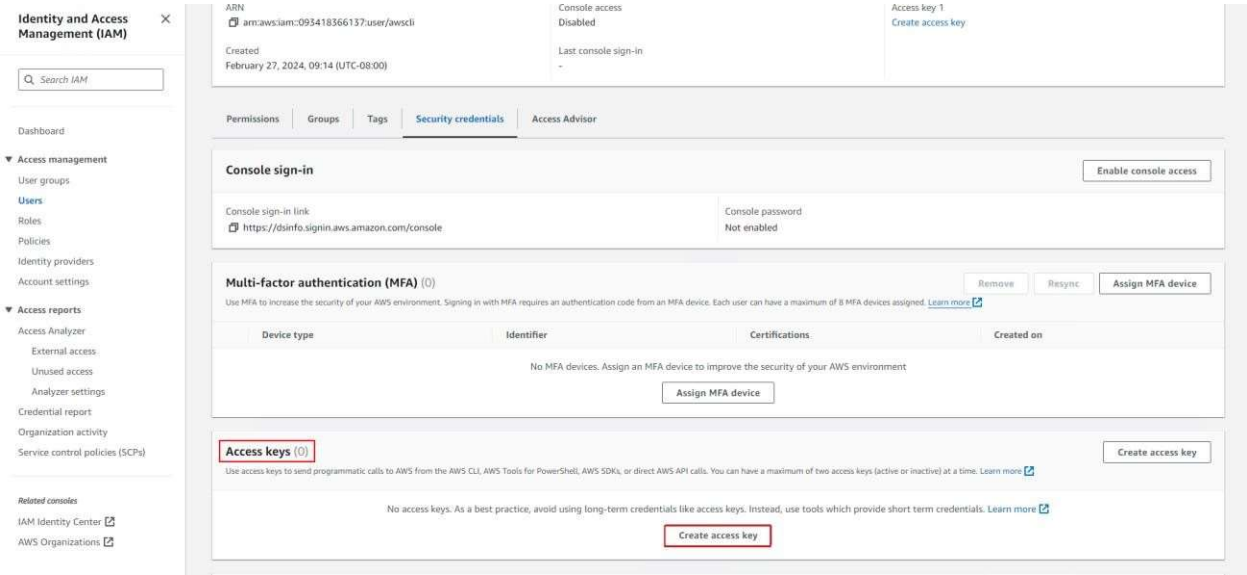
At this stage, select the **Attach policies directly** option, and then choose a **policy** for the user



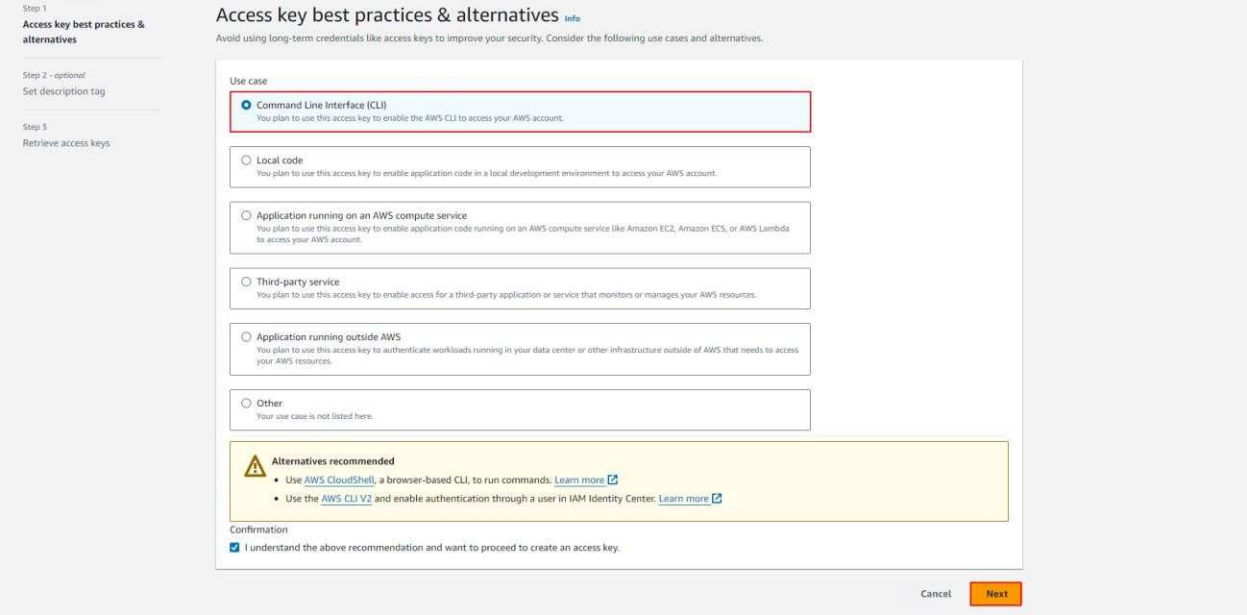
Next, click on the **Create User** button.



After creating the user, select it, and in the **Security Credentials** section, click on the **Create Access Key** button. The **Access Key** acts like a **username** and **password**, and AWS CLI can use it to connect to AWS.

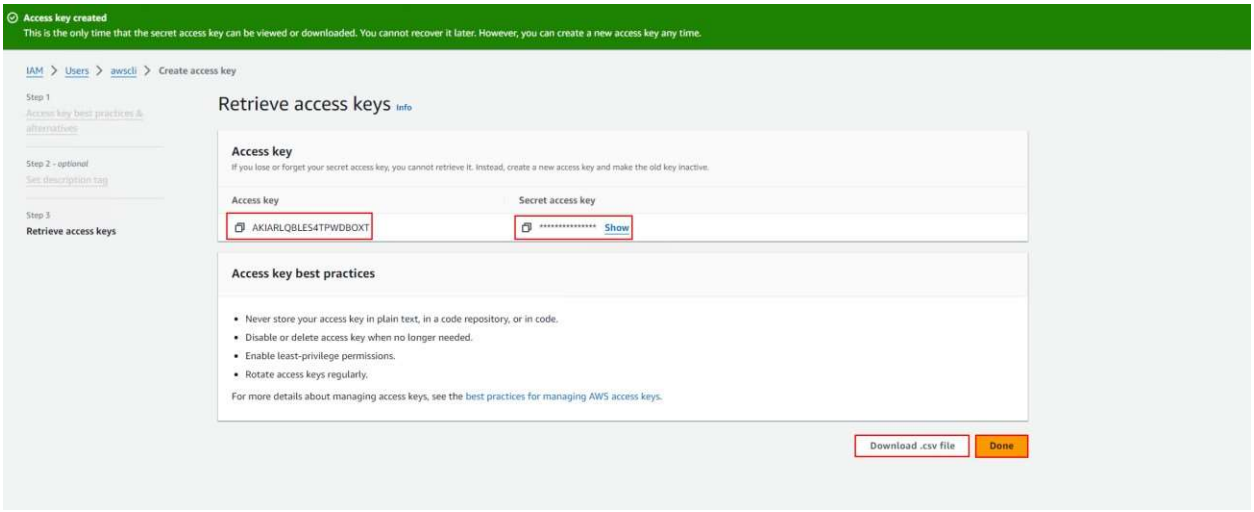


At this stage, select the **Command Line Interface** option, and then click on the **Next** button.

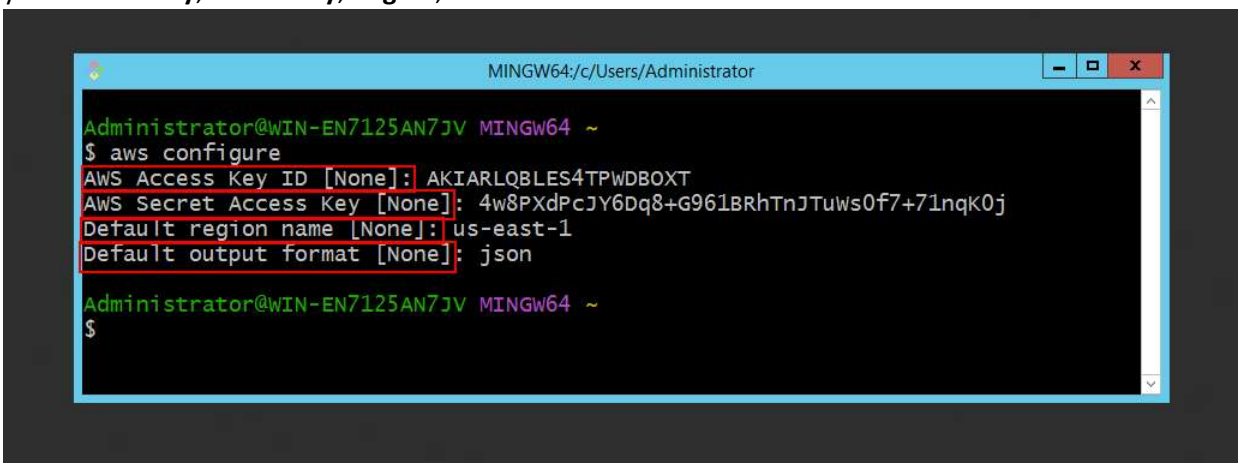


Then, click on the **Create Access Key** button.

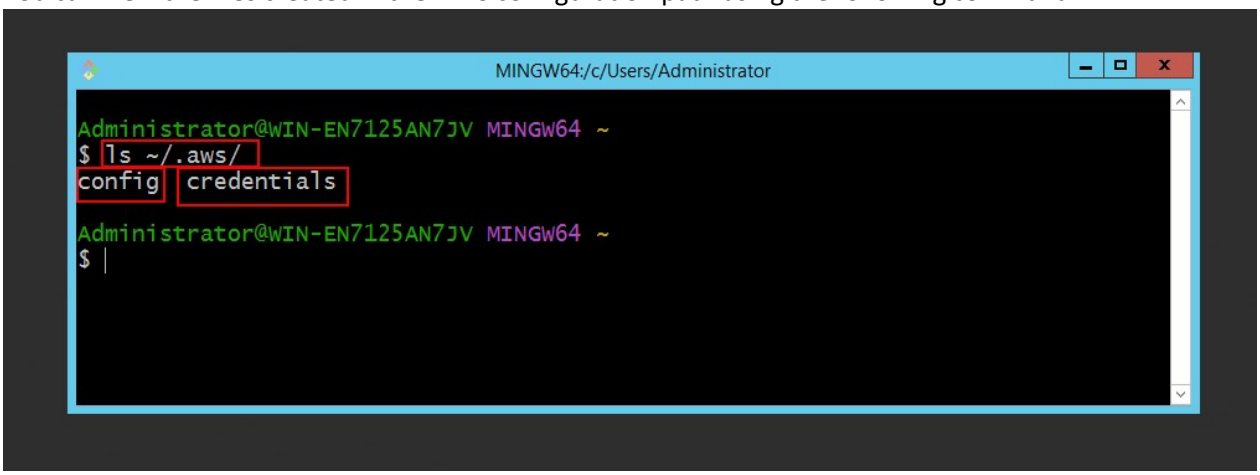
In this section, you will see the **Access Key** and **Secret Access Key**, which are required for logging in through AWS CLI. Click on the **Download .CSV** button and store this information in a secure location.



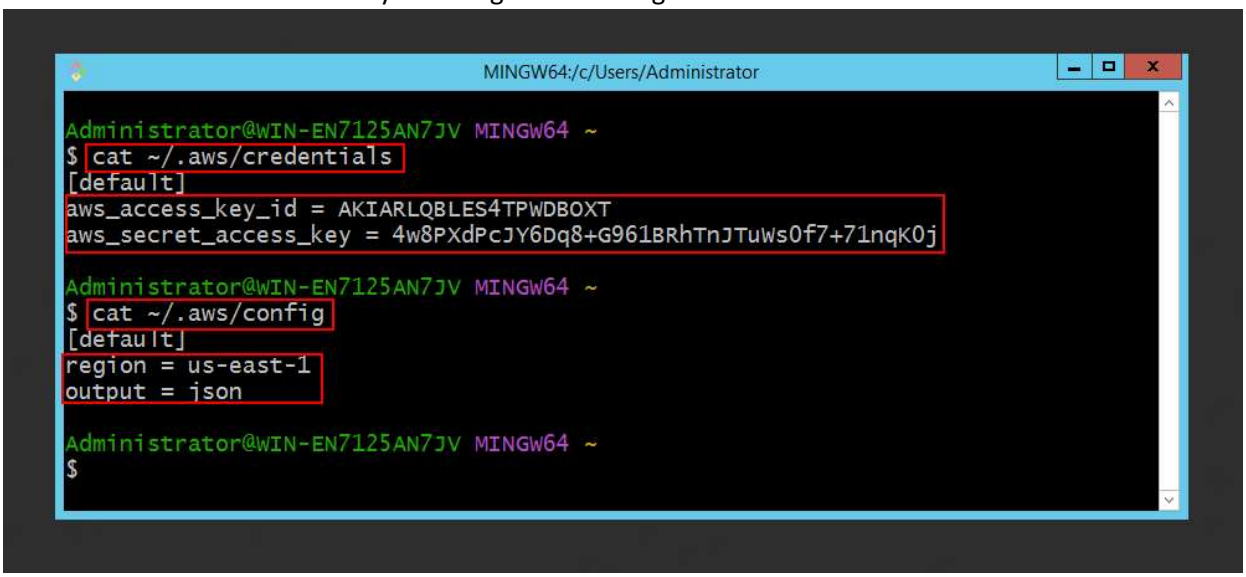
To allow **AWS CLI** to connect to AWS, it needs to be configured. Use the following command to specify your **Access Key**, **Secret Key**, **Region**, and **Data Format**



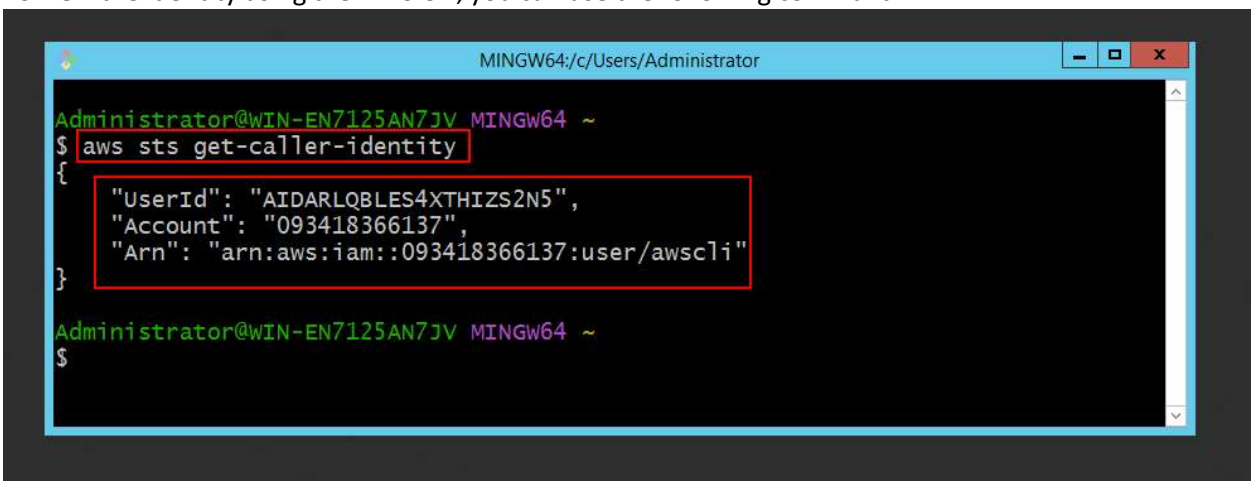
You can view the files created in the AWS configuration path using the following command



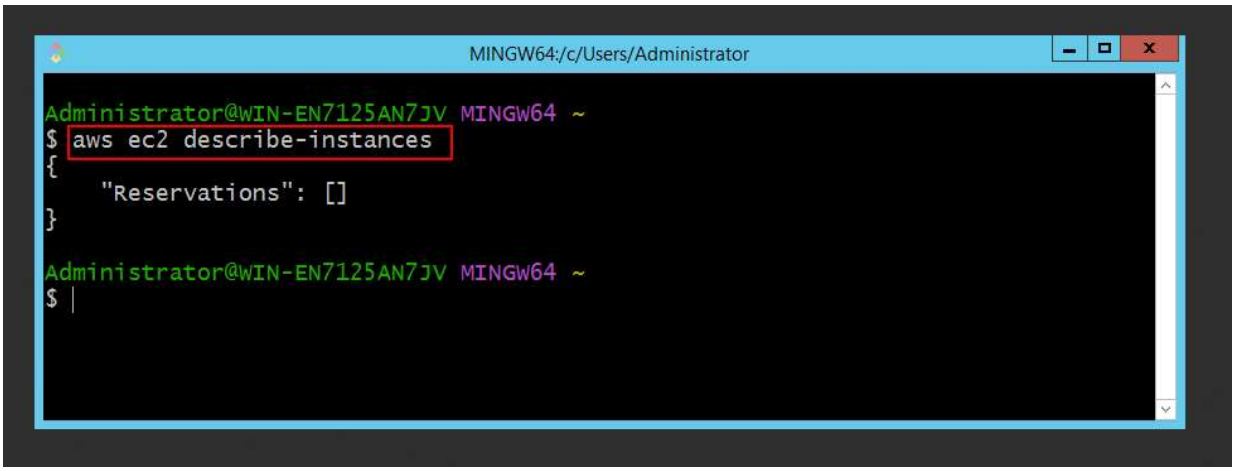
You can view the contents of any file using the following command



To view the identity using the AWS CLI, you can use the following command



To view the details of an EC2 instance, you can use the following command



Introduction to Elastic Block Store (EBS)

EBS is generally a **virtual disk** that can be used with an EC2 instance and has two main purposes: one, it can be used as an **EBS Volume** for virtual disk storage; and two, as a **Snapshot**, which is used to back up an EBS Volume.

Features of EBS

- **Block Storage**, like a hard disk
- Used for **EC2 operating systems** or **data storage** for databases and more.

  For example, the **EC2 EBS Volume** serves as the **Root Volume**, where the **operating system data** is stored.

You can also store other data such as **databases**, **web server content**, or any other files on it.

- EBS resides within an **Availability Zone**, enabling **replication across zones**, which increases overall **availability**.
- Use of **Snapshots** to **back up** EBS Volumes.

Types of EBS

You can choose from different types of EBS based on **price** and **performance**, including the following:

- **General Purpose SSD (gp2 / gp3)**

  This type of EBS is used for a variety of workloads, such as running a **web server** on it.

- **Provisioned IOPS (io1 / io2)**

  It is used for running **large databases** and provides **high performance** for your database workloads.

- **Throughput Optimized HDD (st1)**

  This type of EBS is used for **Big Data** and **Data Warehouse** workloads.

- **Cold HDD (sc1)**

  It is generally used for setting up a **file server** and has a very **low cost**.

-Magnetic

They have **low performance** and are mainly used for **backups** and **archiving data**, with a very **low cost**.

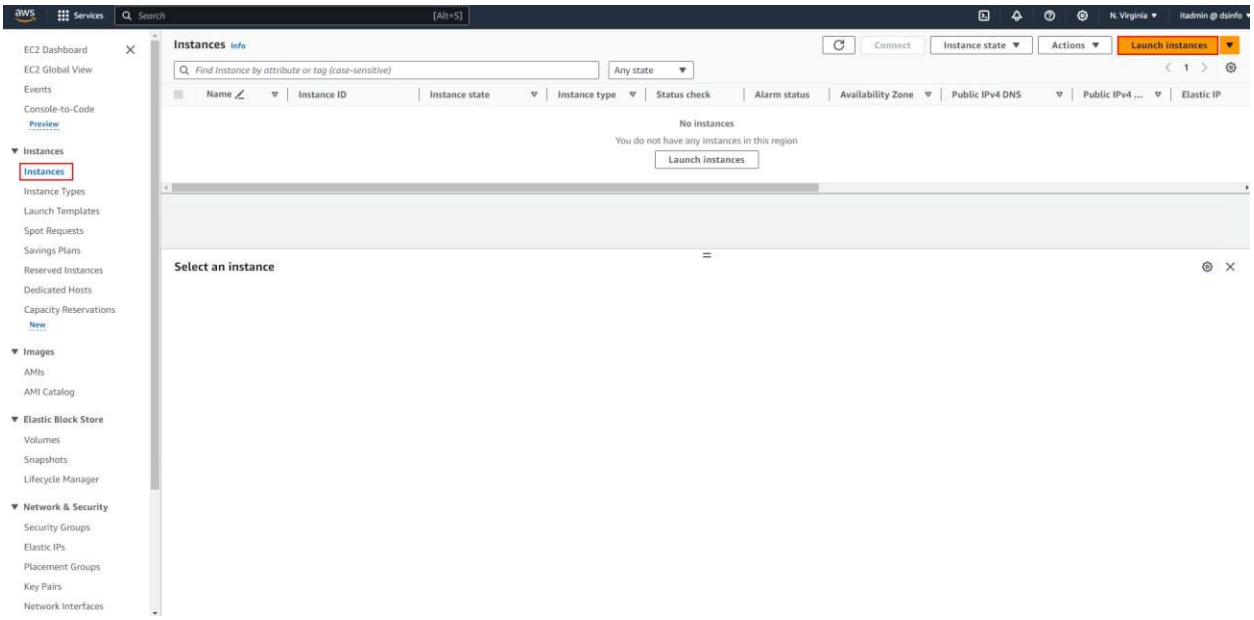
How to Create an EBS Volume

Before creating an EBS Volume, we first want to launch an **EC2 instance** with the **CentOS** operating system. Prior to launching, we will use a **script in the User Data section** so that a web server is automatically set up and a website is deployed on it. In other words, we want the web server setup to happen **automatically during instance creation**, instead of doing it **manually** afterward.

To complete this scenario, you need to follow these steps:

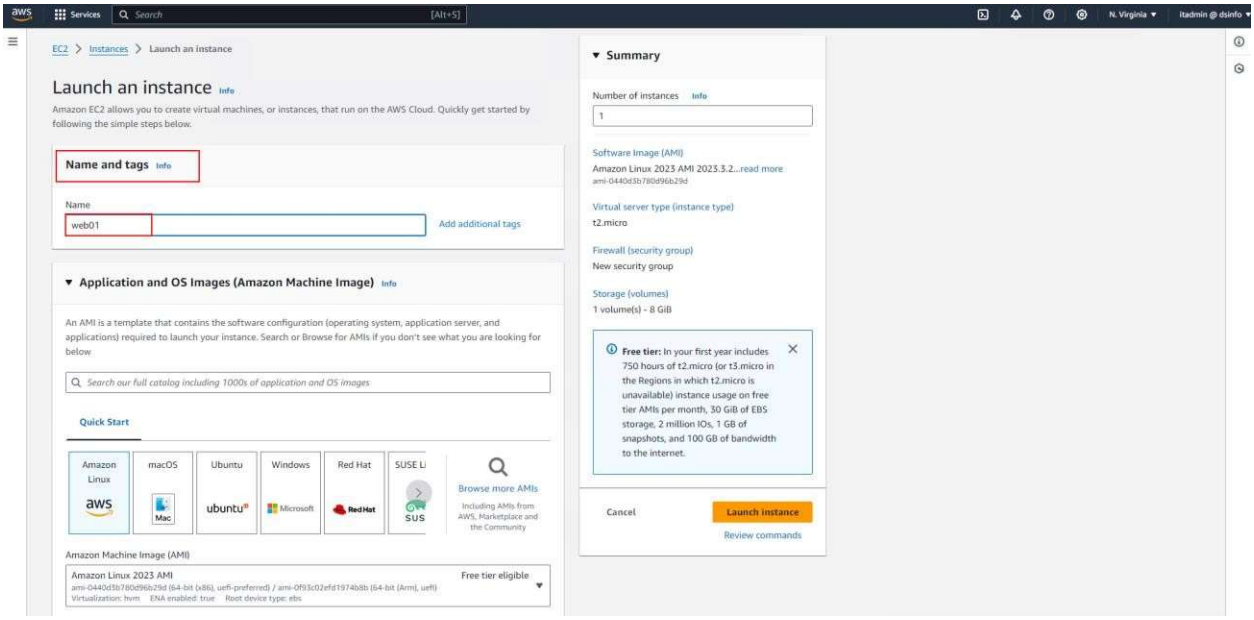
Step 1:

In the **Instances** section, click on the **Launch Instances** button.



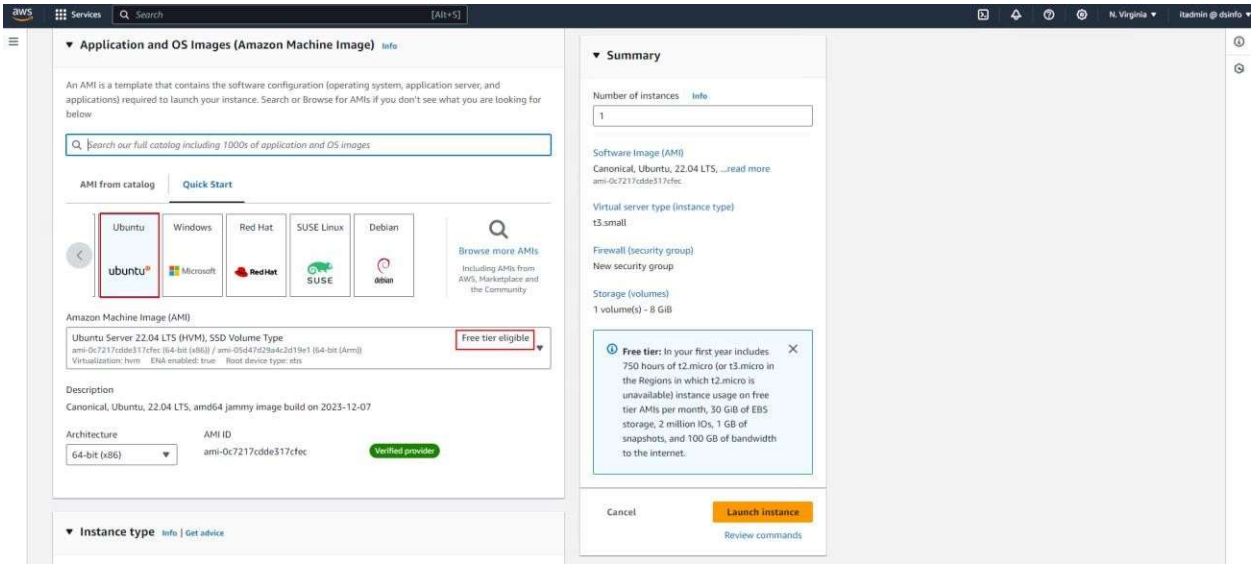
Step 2:

In this step, specify a name in the **Name** section, for example: **web01**.

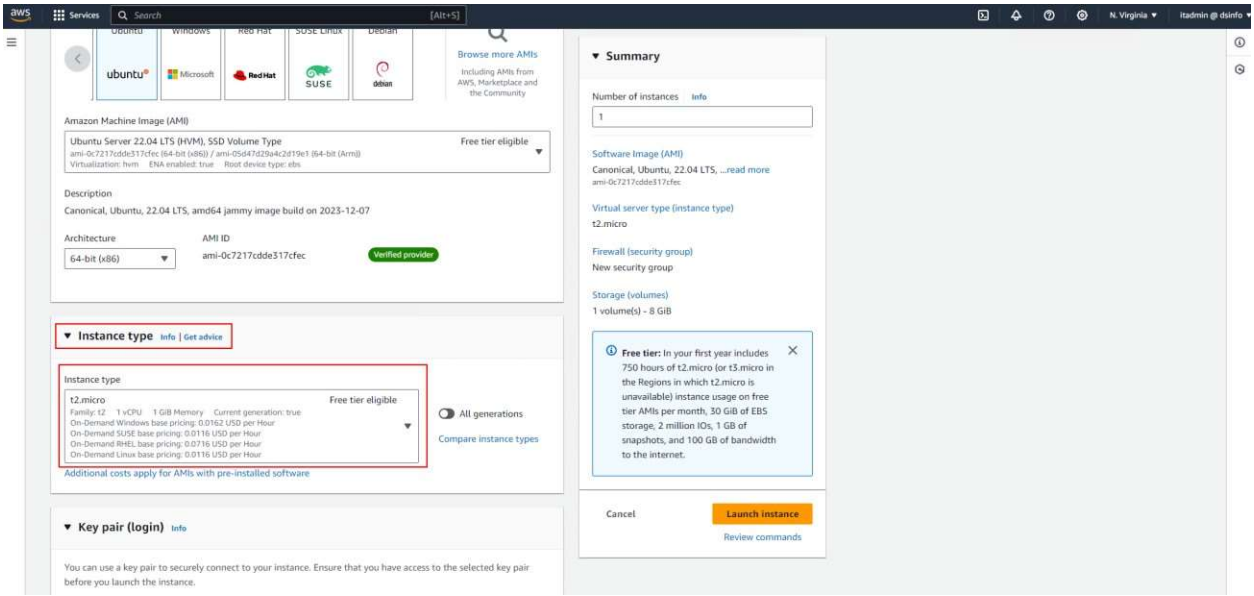


Step 3:

In the **AMI** section, select the **Ubuntu** image.

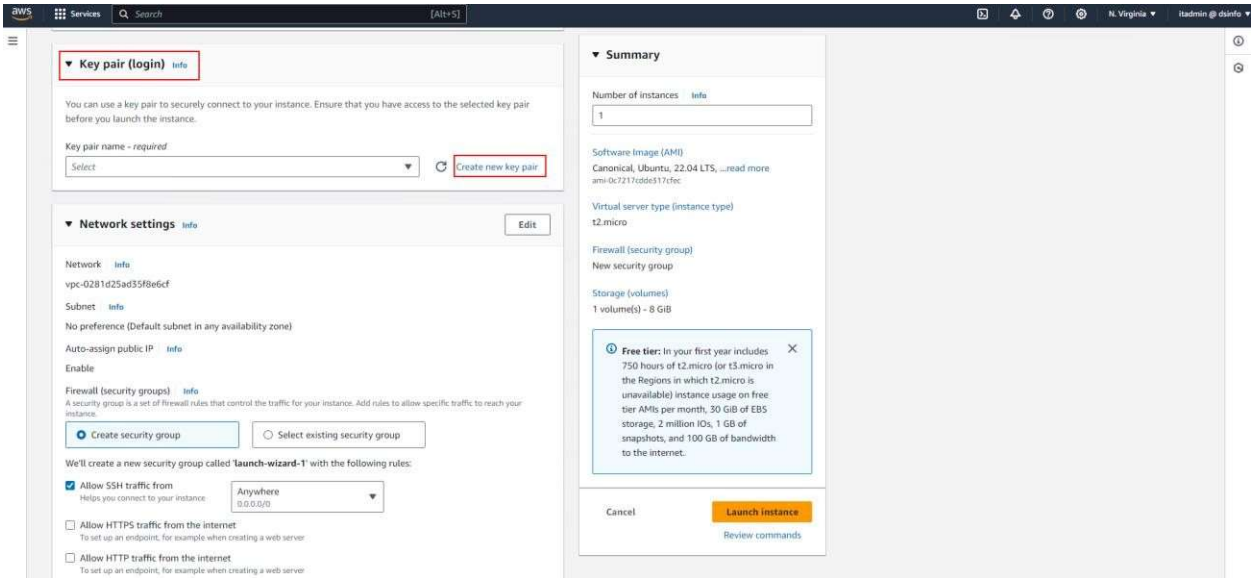


At this stage, in the **Instance Type** section, select **t2.micro**.

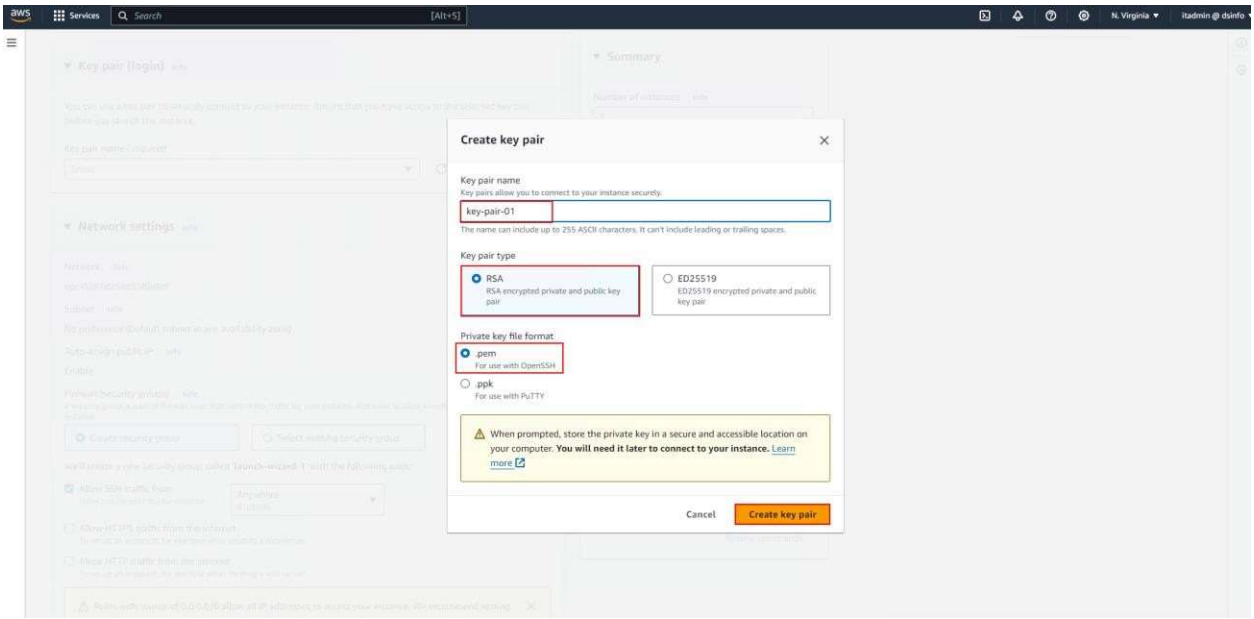


Step 4:

In the **Key Pair** section, click on the **Create new key pair** link and create a key pair for login.

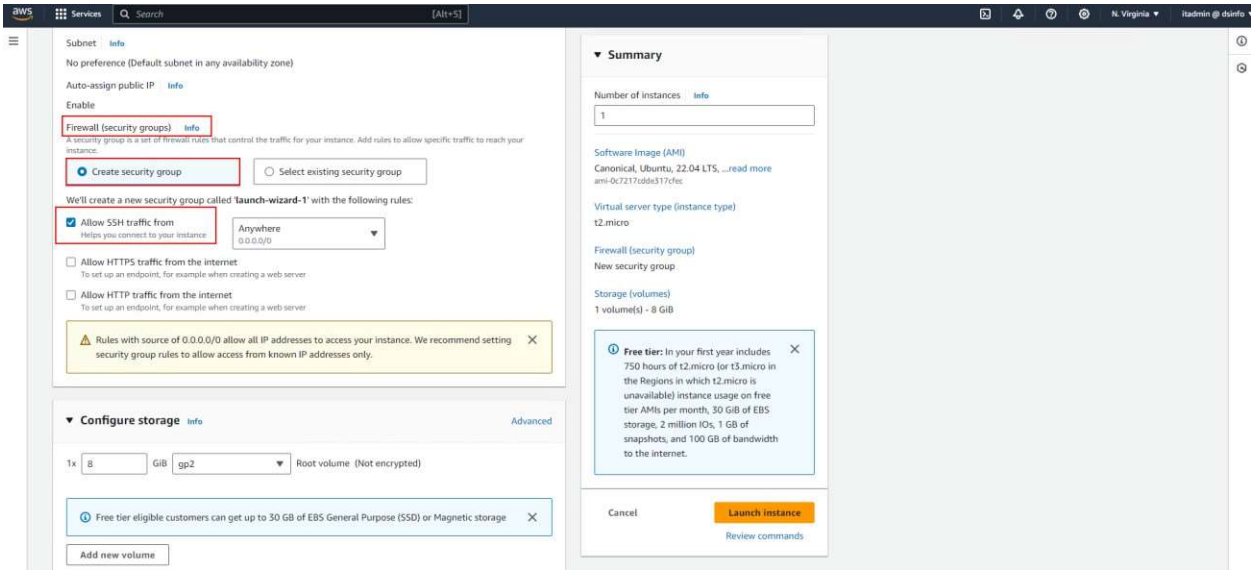


At this stage, create a Key Pair with the following specifications.

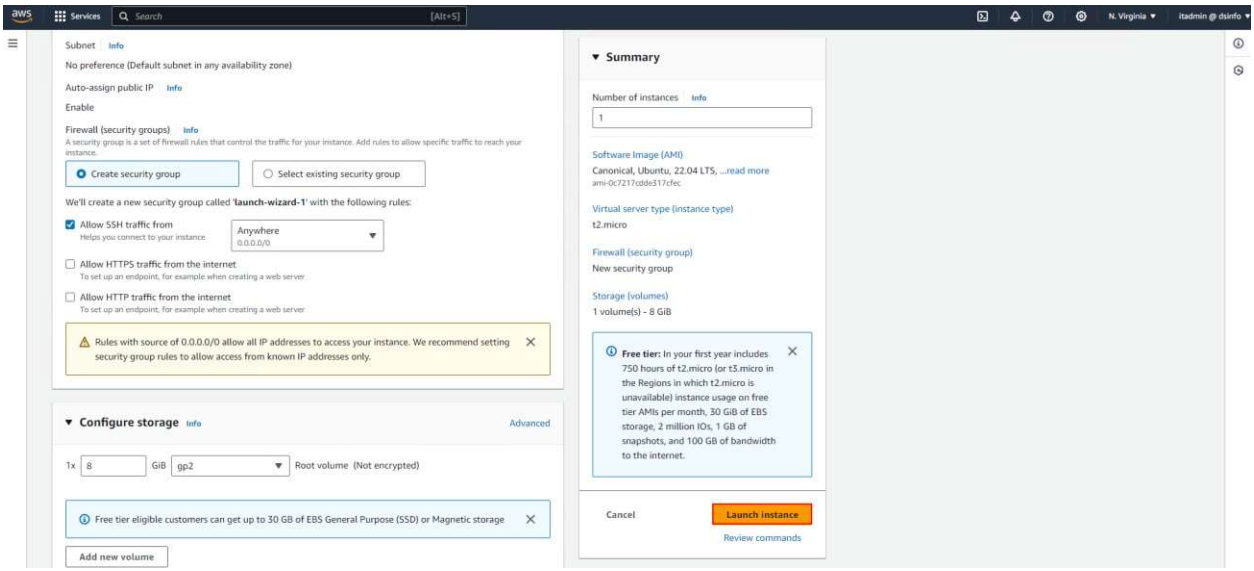


Step 5:

In the **Security Group** section, select the **Create Security Group** option.

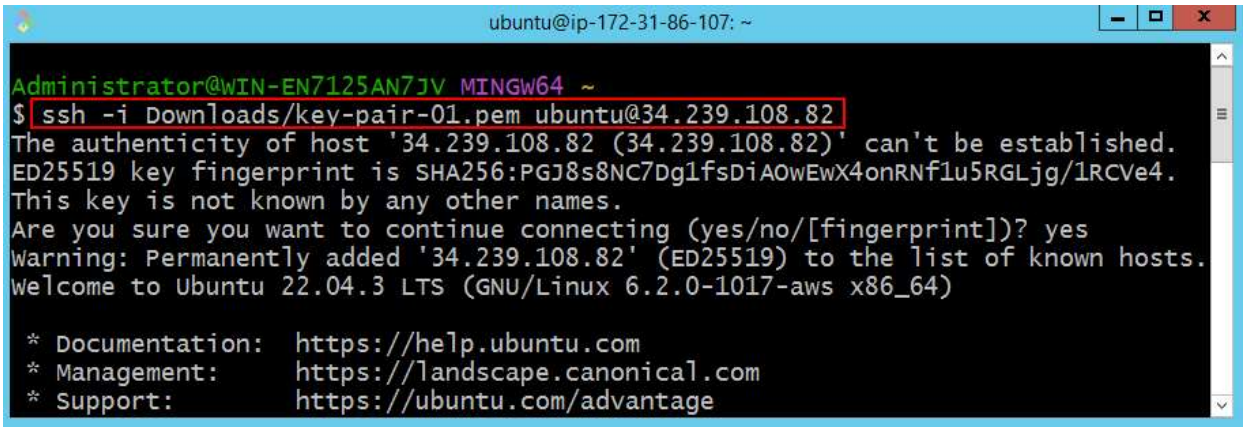


Then, click on the **Launch Instance** button to create the instance.



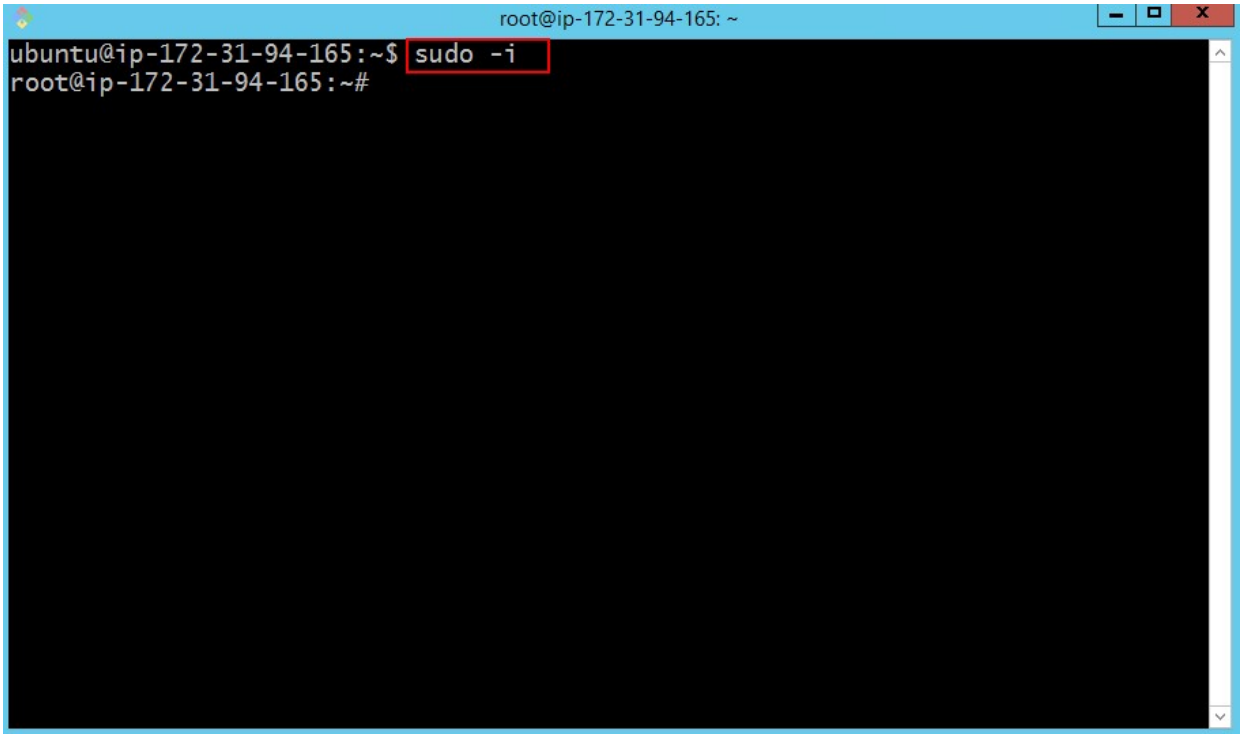
Step 6:

At this stage, connect to the instance using **SSH** and run the following commands to set up the web server on the instance.

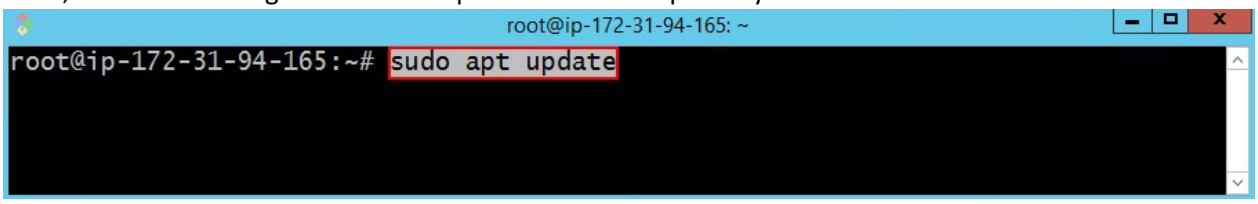


Now, in this section, we want to install the **Tween** website, which is one of the templates from **Tooplate**, on this EC2 instance.

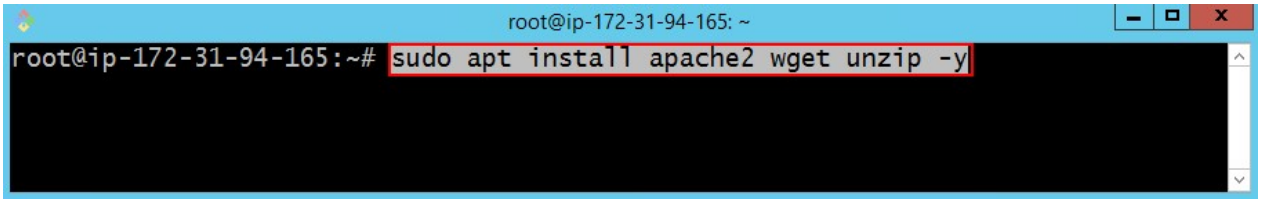
At this stage, use the following command to switch to the root user environment



Then, use the following command to update the Linux repository

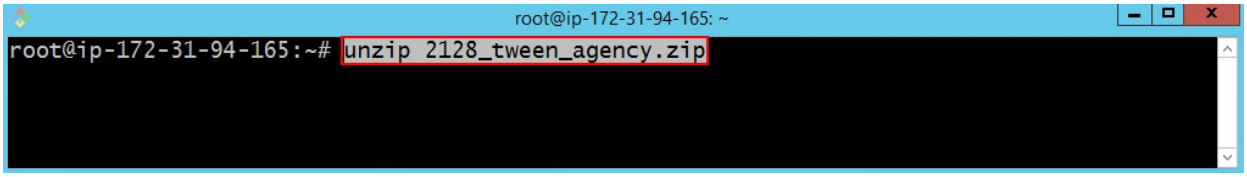


At this stage, use the following command to install the **Apache**, **Wget**, and **Unzip** packages

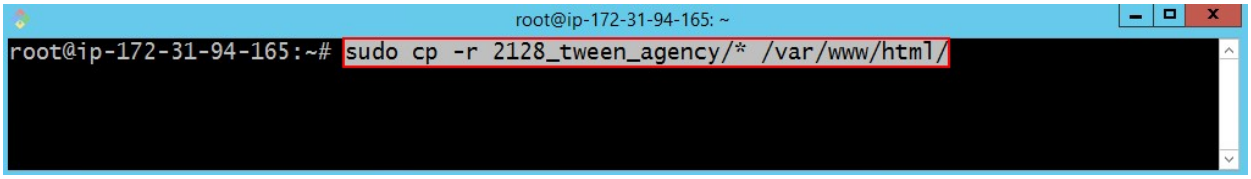


Next, you need to download the **Tween** file from the **Tooplate** website.

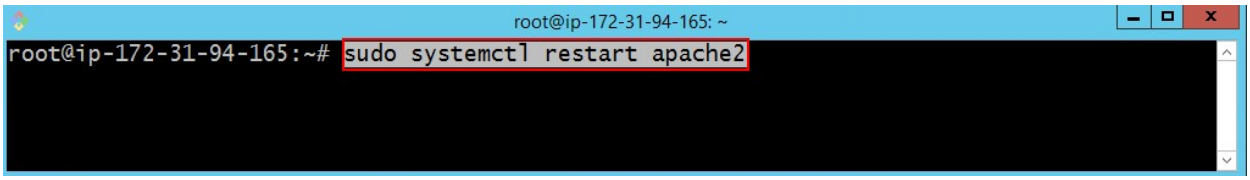
After downloading the **Tween** file, you need to unzip it.



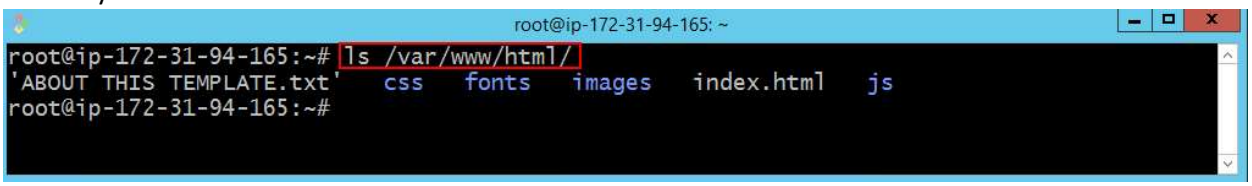
At this stage, you need to copy the unzipped contents of the **Tween** website into the **web server directory**.



And finally, you need to restart the **Apache** service.

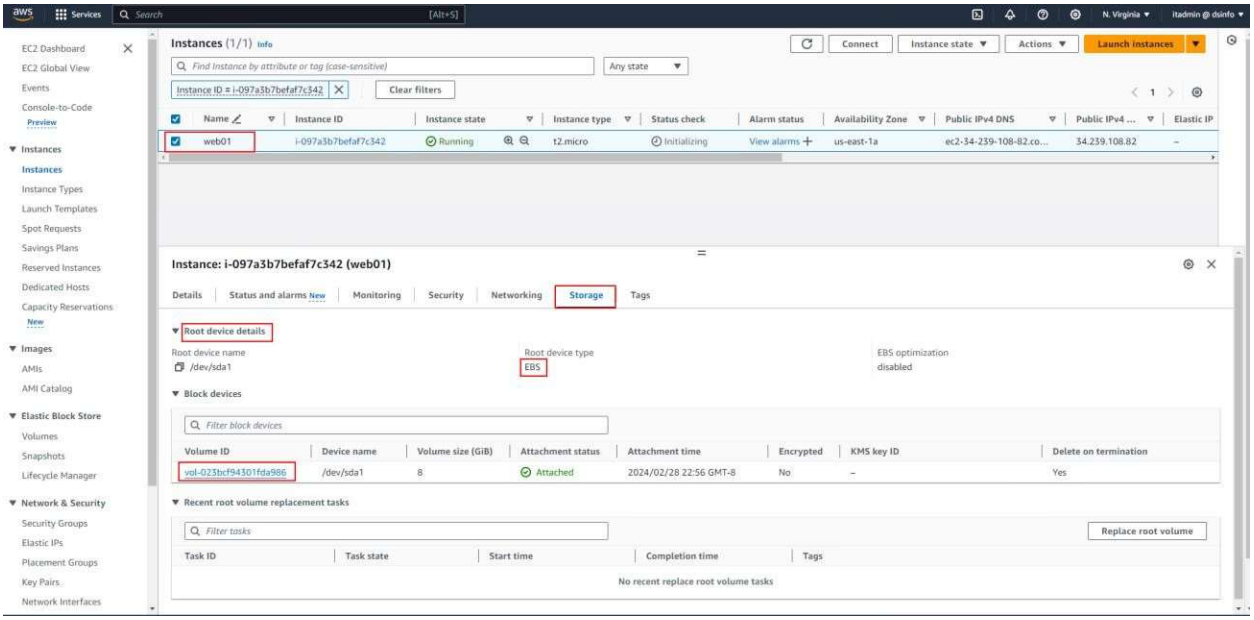


Use the following command to check if the **Tween** website files have been copied to the web server directory

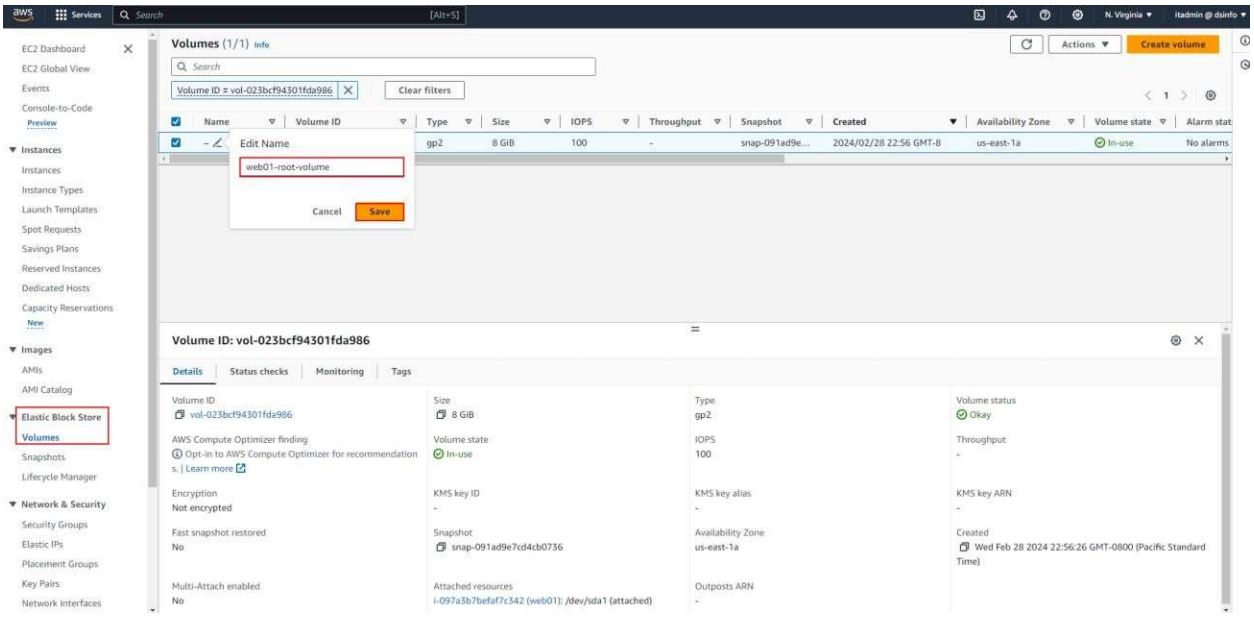


Now, we intend to move the **image** folder to a separate **EBS Volume**.

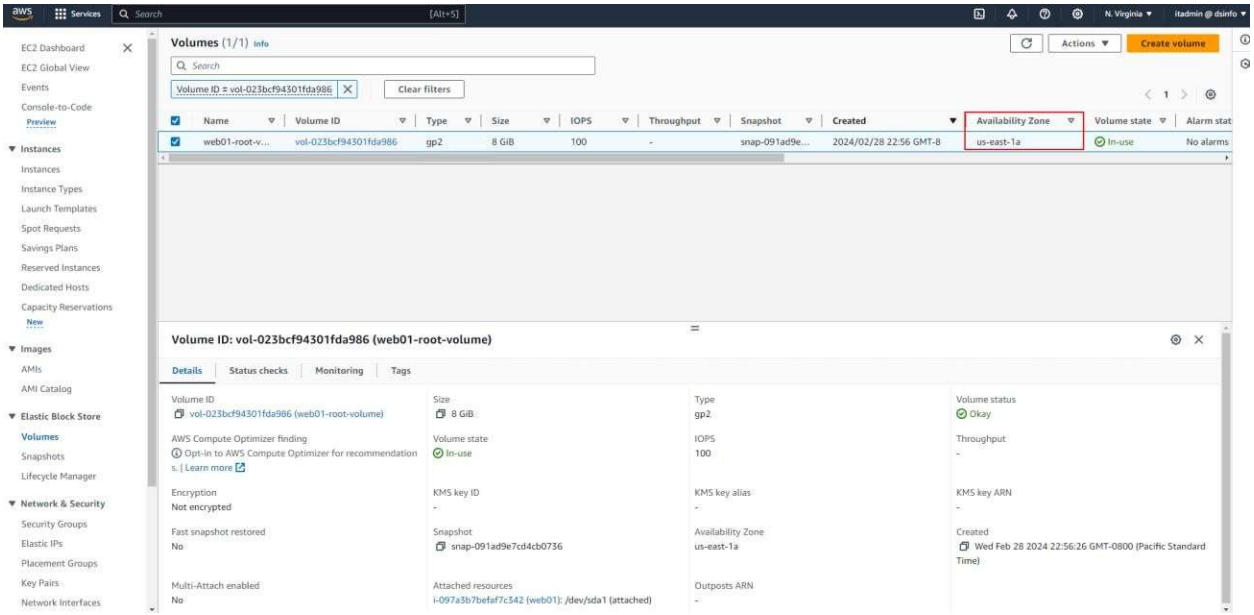
If you click on **Storage** in the **Instance** section, you can view details about the **Root EBS Volume**. By clicking on the **Volume ID**, you will be taken to the **EBS Volume** section.



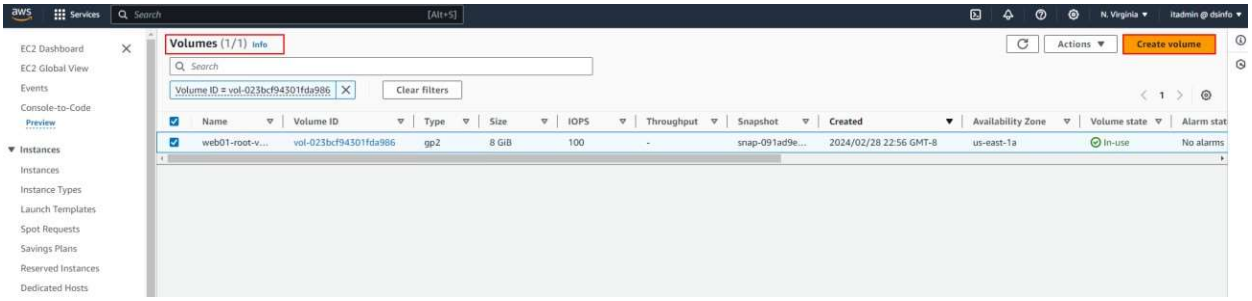
In the **EBS Volume** section, click on the **Name** field and set it to **web01-root-volume**.



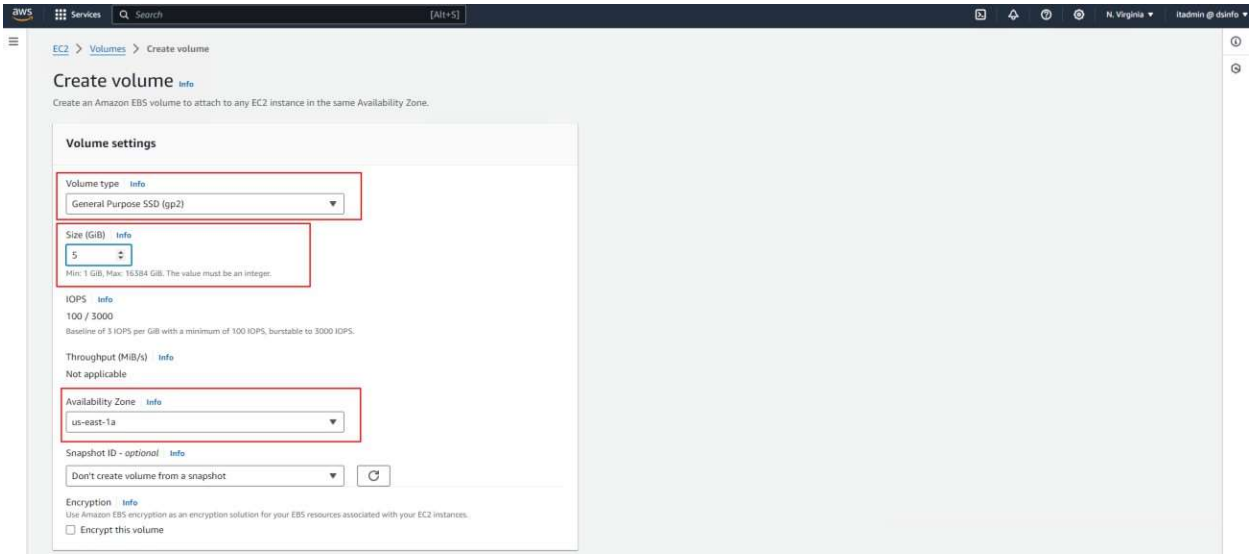
In the **Availability Zone** section, as you can see, the zone of the **Volume** and the **Instance** is the same — in other words, they must be in the **same zone**.



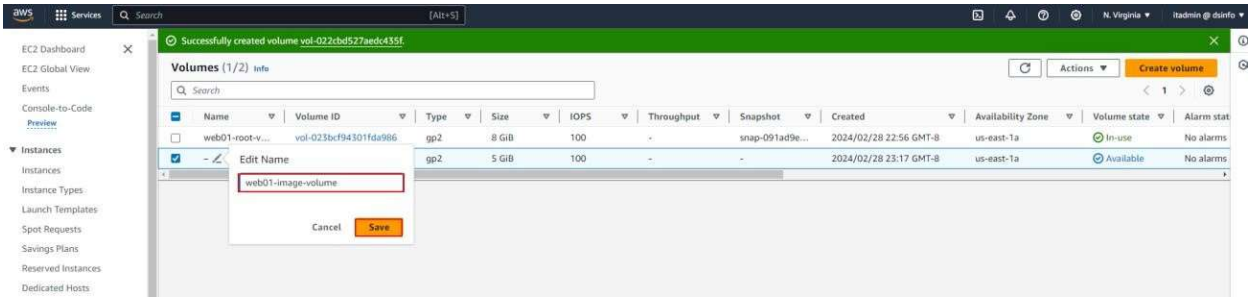
To create a volume, click on the **Create Volume** button.



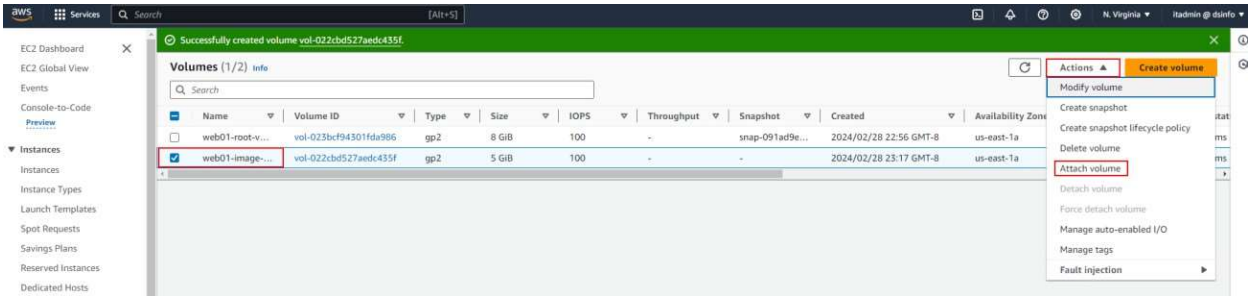
At this stage, select **General Purpose SSD (gp2)** from the **Volume Type** section, set the **Size** to **5GB**, and make sure to choose the same **Availability Zone** as your instance. Finally, click the **Create Volume** button.



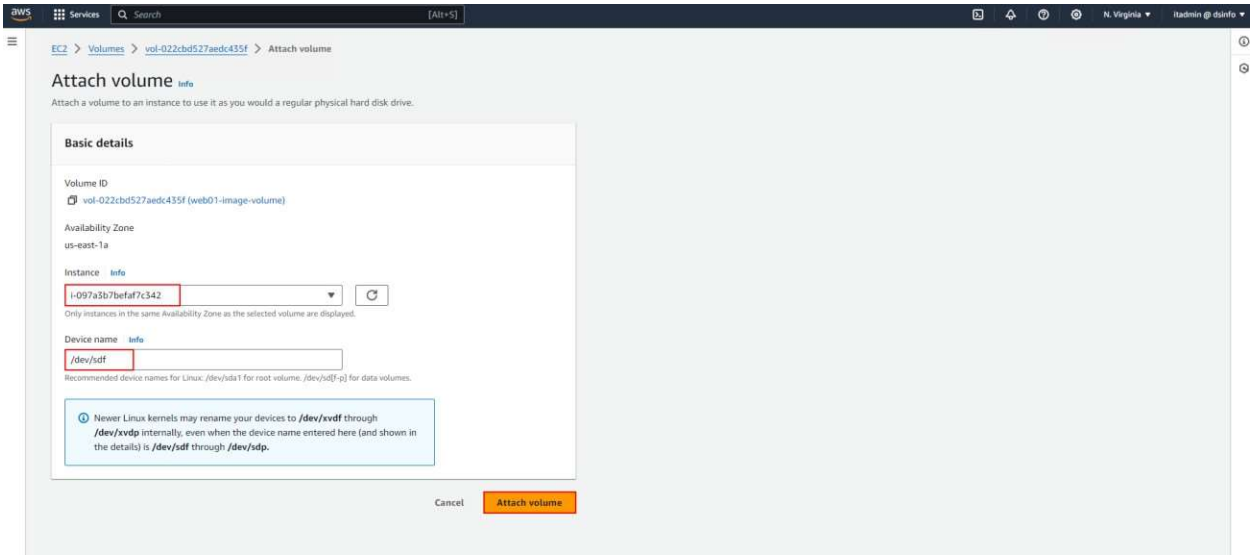
At this stage, assign a **name** to the EBS Volume.



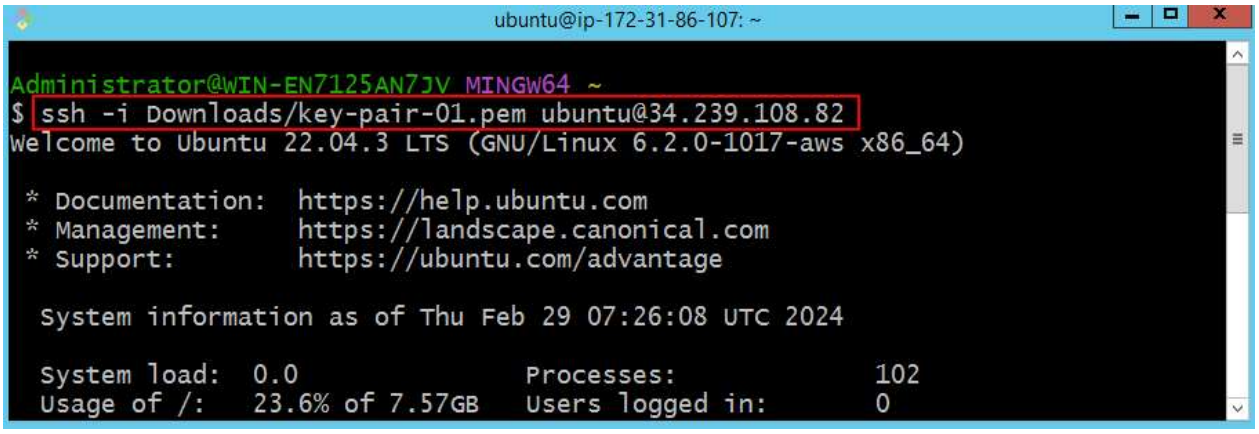
To attach the **EBS Volume** to the **EC2 Instance**, click on the **Actions** button and then select **Attach Volume**.



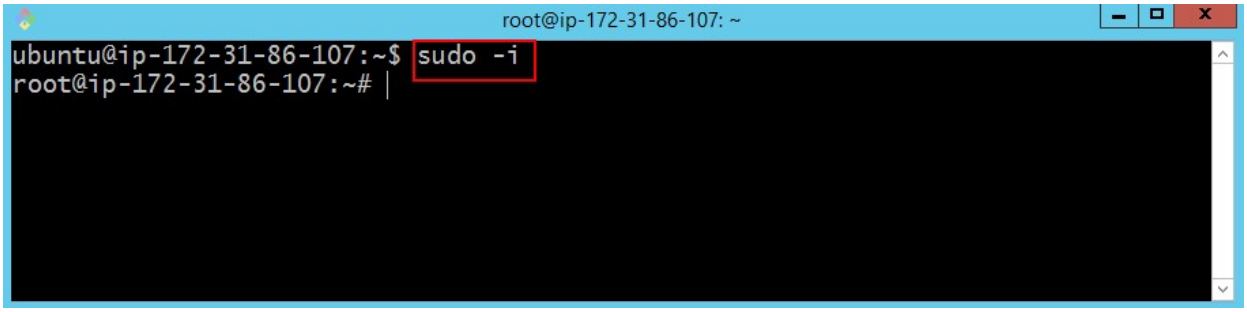
Then, at this stage, select the **Instance name** and click the **Attach Volume** button.



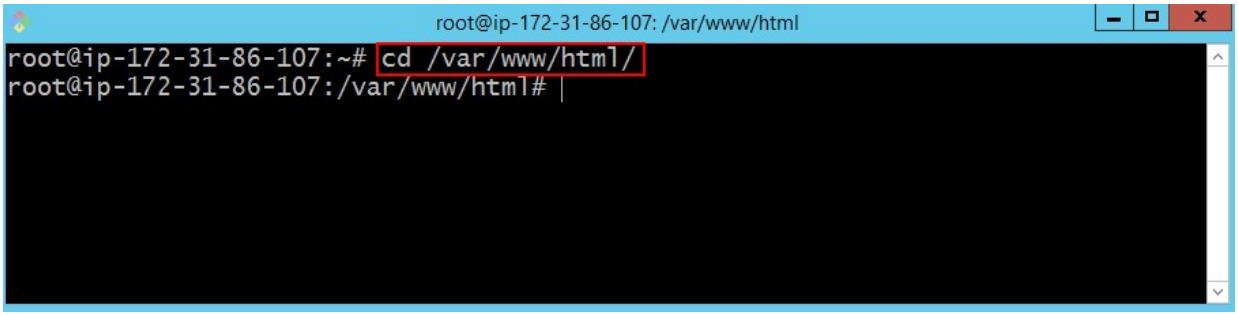
After attaching the **Volume** to your instance, you can log in to your instance.



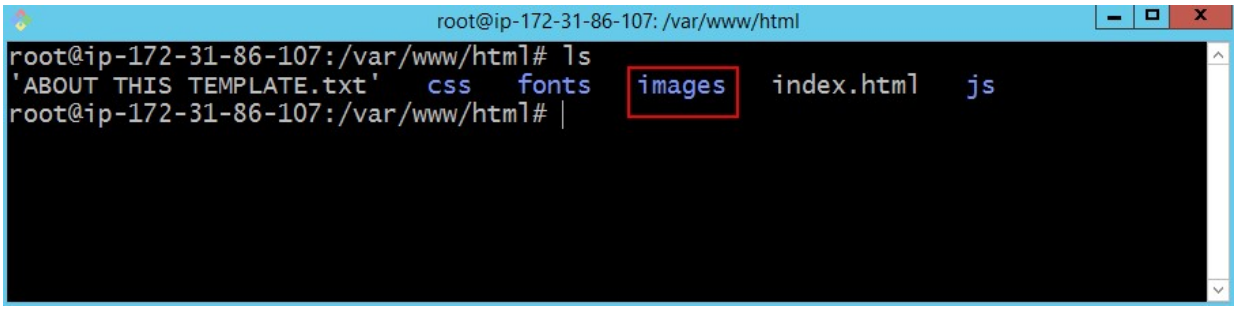
After logging in, use the following command to switch to the **Root User**



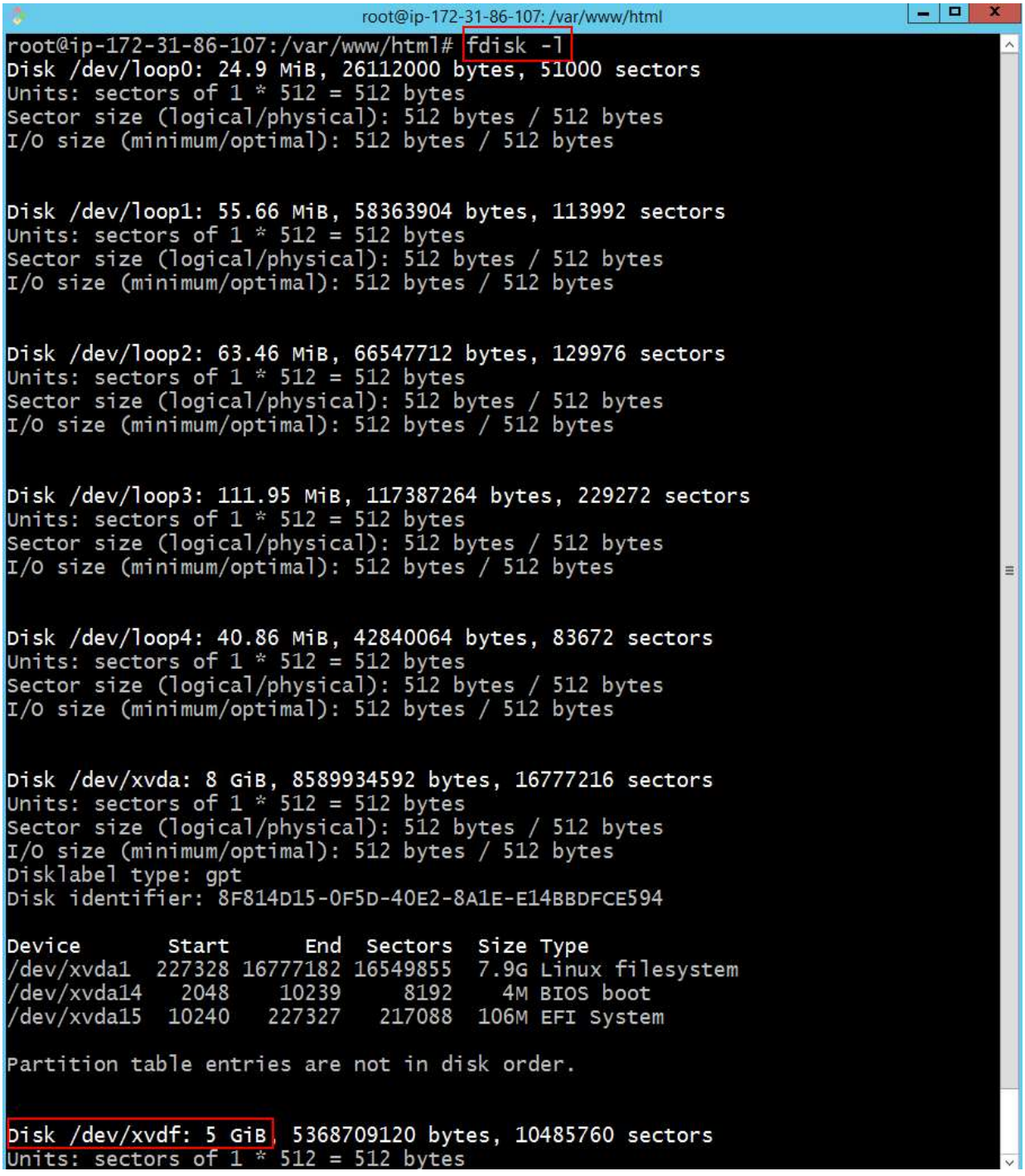
Enter the **web server directory**



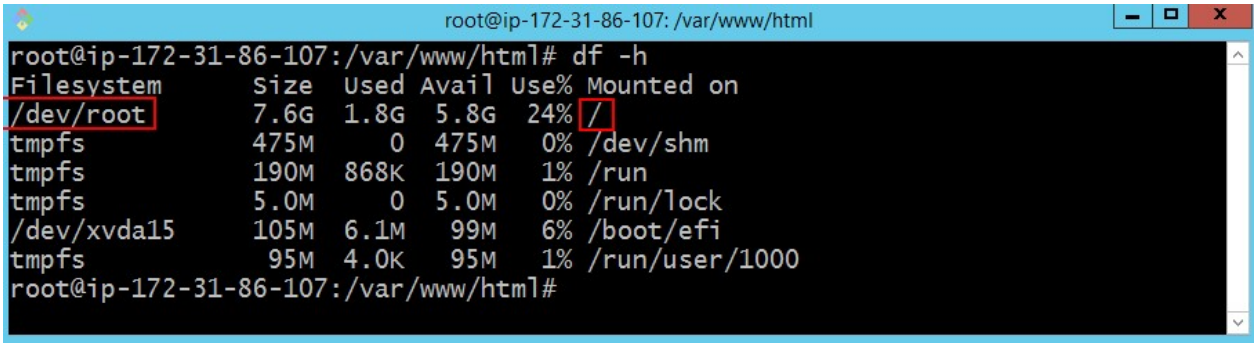
We intend to move the **image** folder to a **different storage (EBS volume)**.



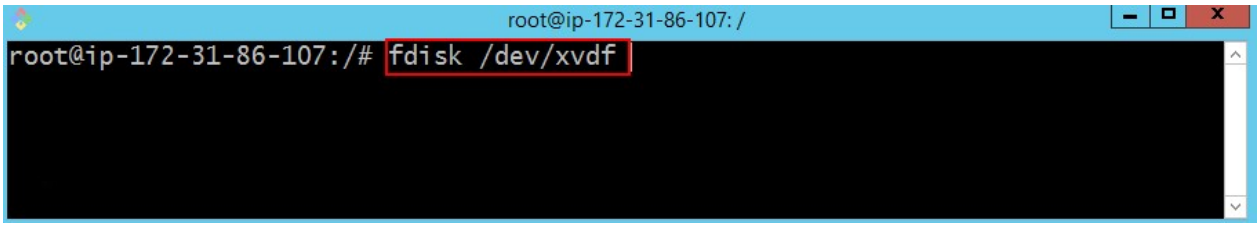
First, use the following command to view the list of available partitions



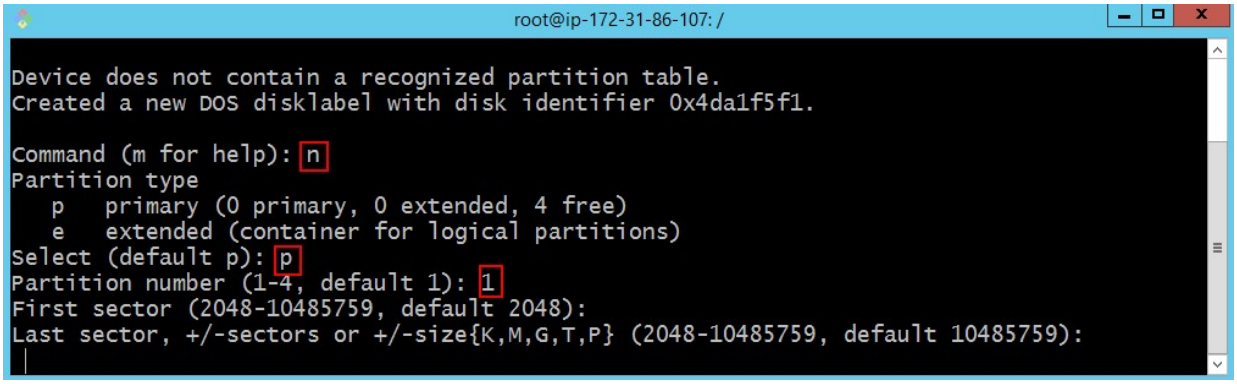
Use the following command to view the mounted partitions



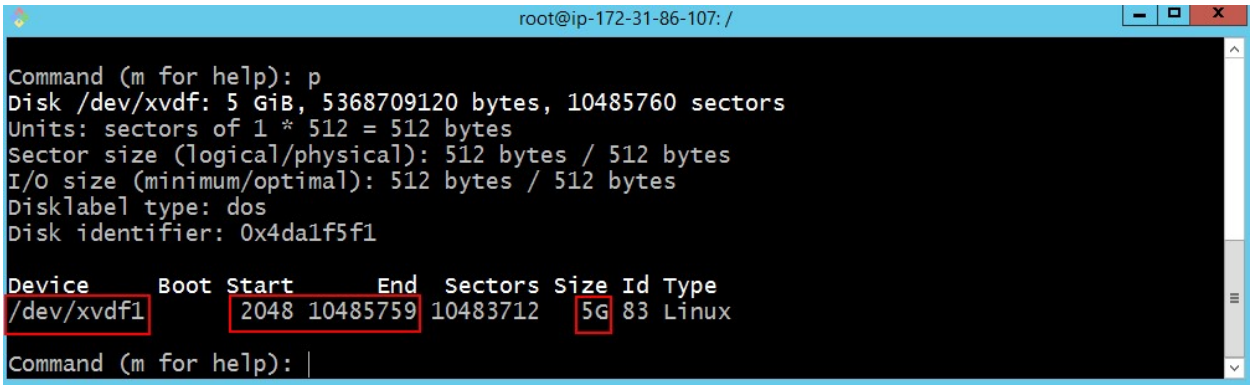
At this stage, you need to partition the new volume using the following command



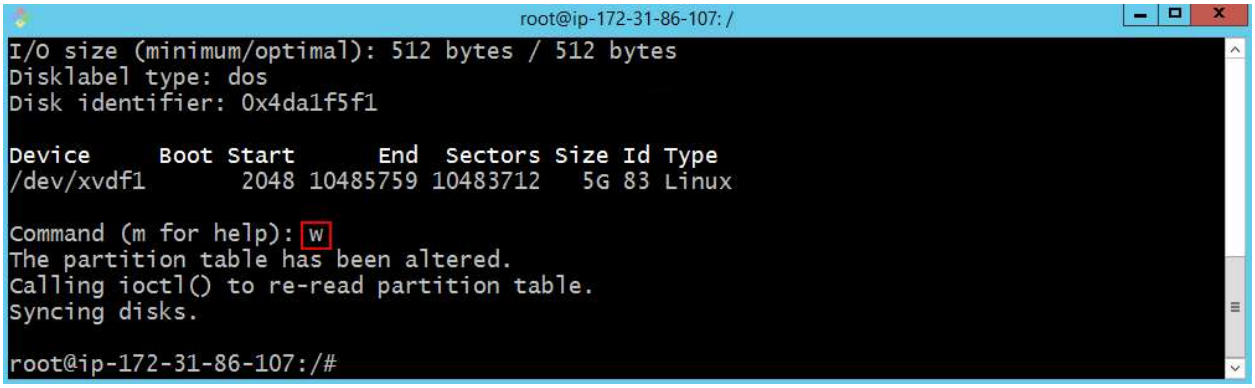
Next, press the **m** key to view the **fdisk help menu**. To create a new partition, press **n**, then **p** for Primary Partition, set the partition number to **1**, and press **Enter**.



To view the created partition, simply press the **p** key, which stands for **print**.

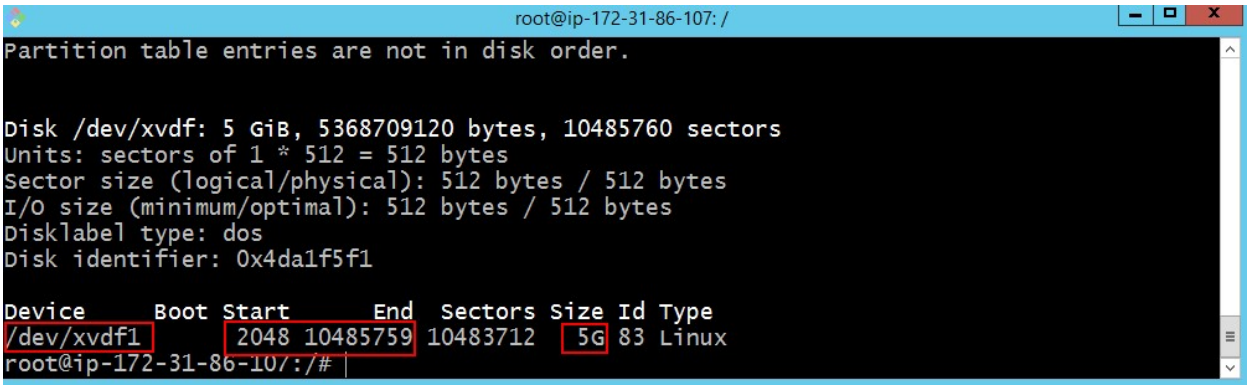


To **write** the changes, press the **w** key.

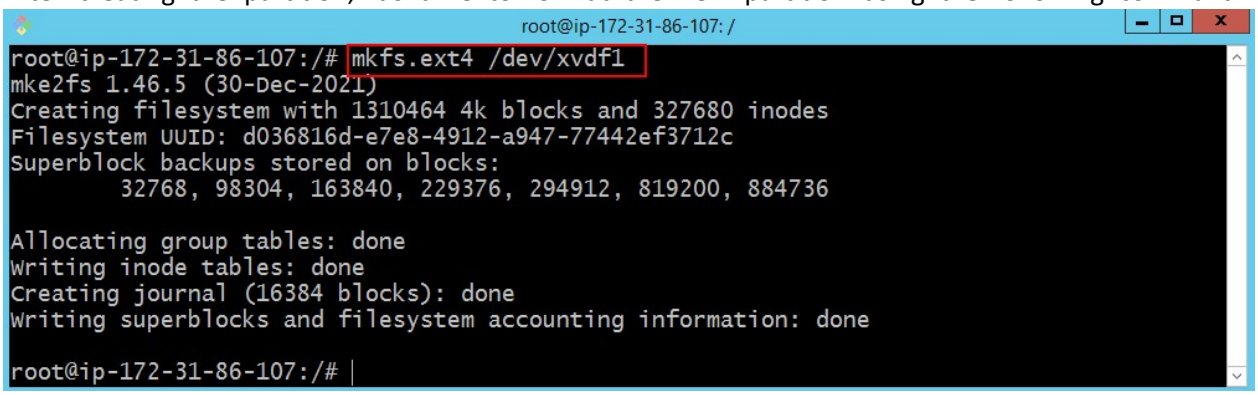


To view the created partition, you can use the following command:

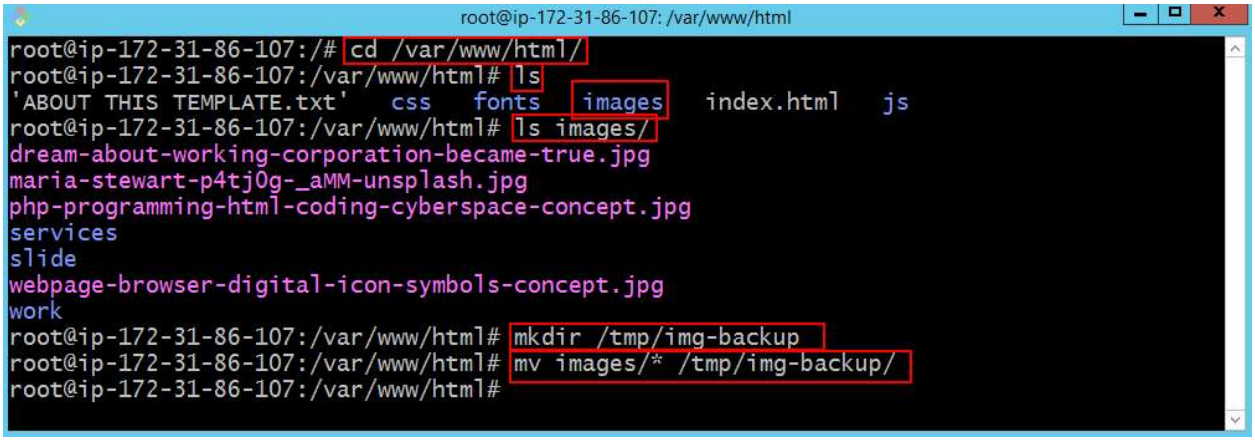
```
fdisk -l
```



After creating the partition, it's time to format the new partition using the following command

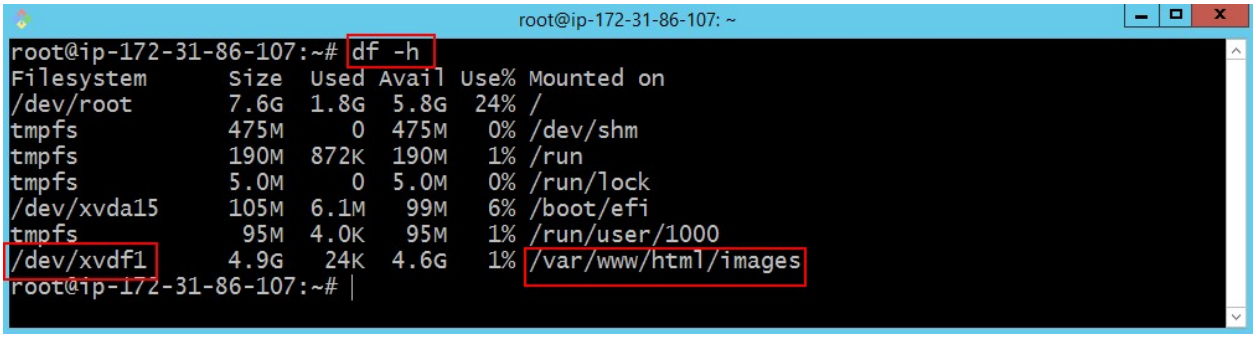


Now, to mount the **image** folder, go to the **web server directory** and review the contents of the **image** folder. Then, create a folder in the **/tmp** directory and back up the contents of the **image** folder to avoid data loss during the mount process.

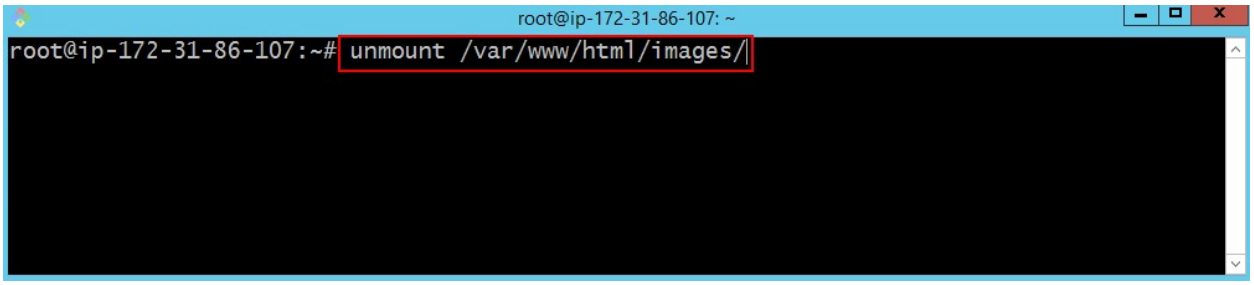


Now, at this stage, use the following command to mount the new partition to the **images** folder in the web server directory

To check whether the mount was successful, use the following command



If you want to unmount the volume, you can use the following command

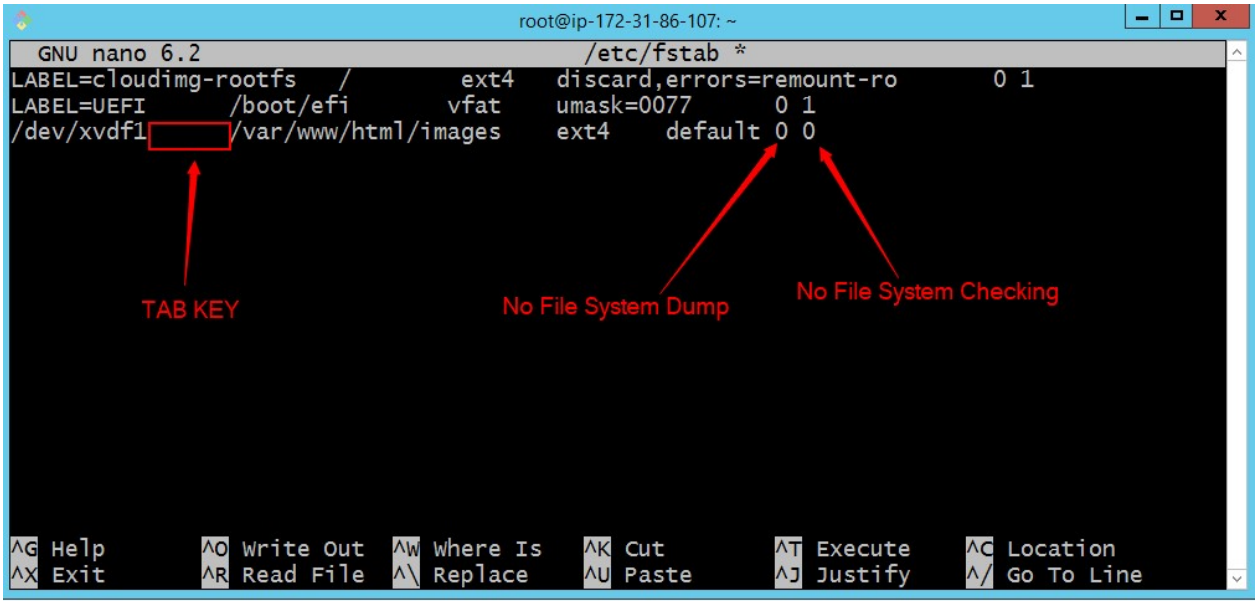


One important point to note is that if the instance is stopped and started, or in other words **restarted**, the mounted partition will be **unmounted**.

For **permanent mount**, the mount configuration must be added to the **fstab** file.

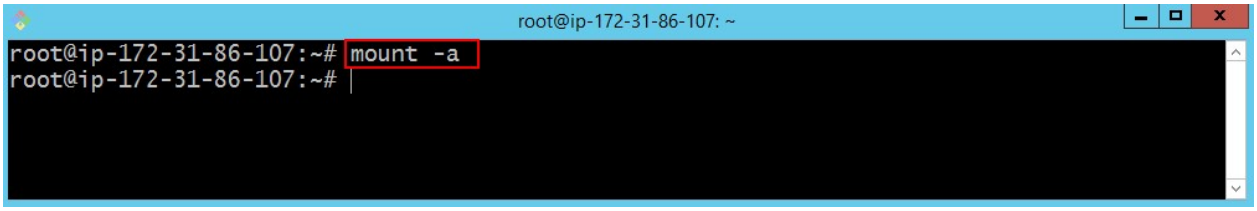
So, open the **fstab** file using the **nano** editor

Now, inside the file, enter the mount information as shown below

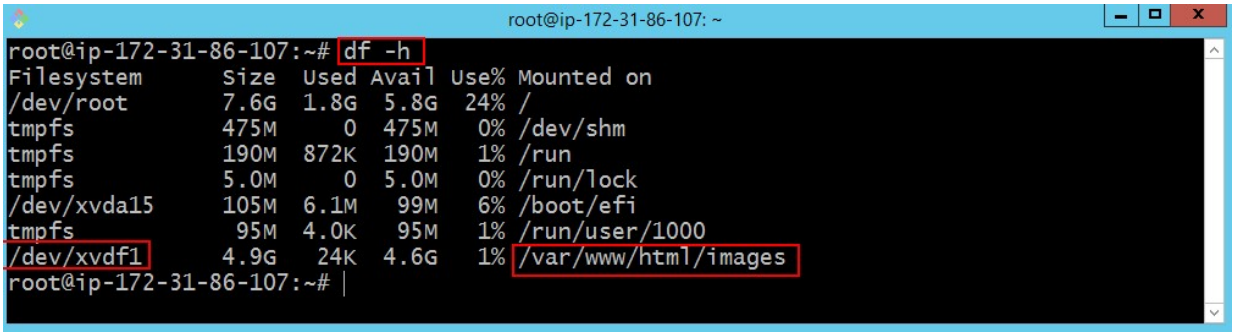


Then, **save** the file.

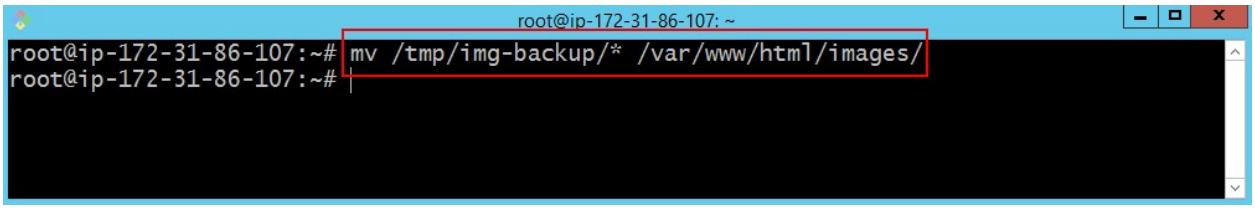
Use the following command to mount all entries defined in the **fstab** file



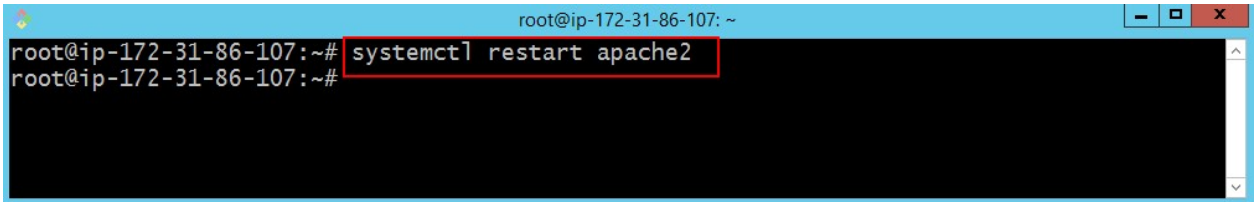
Again, use the following command to ensure that the mount has been done correctly



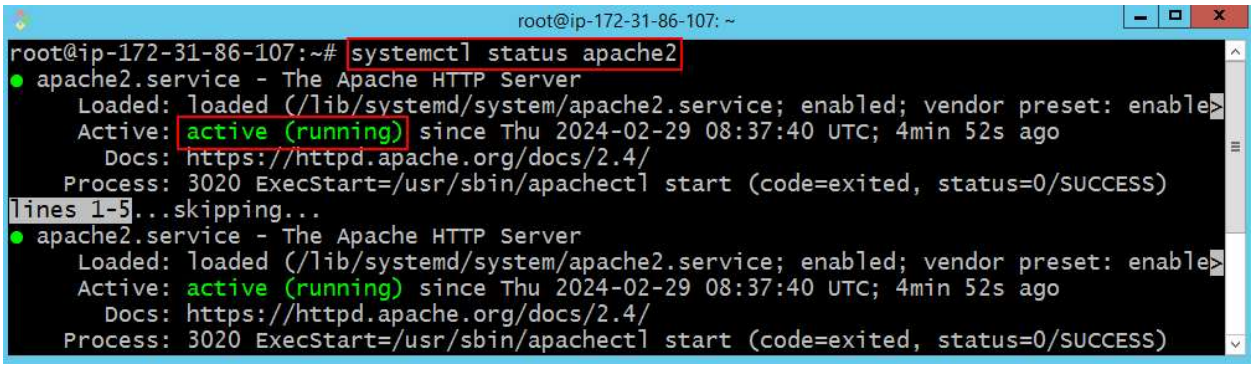
At this stage, move the files you backed up in the **/tmp** directory back to the web server's **images** folder, which is now located on the new volume. Use the following command



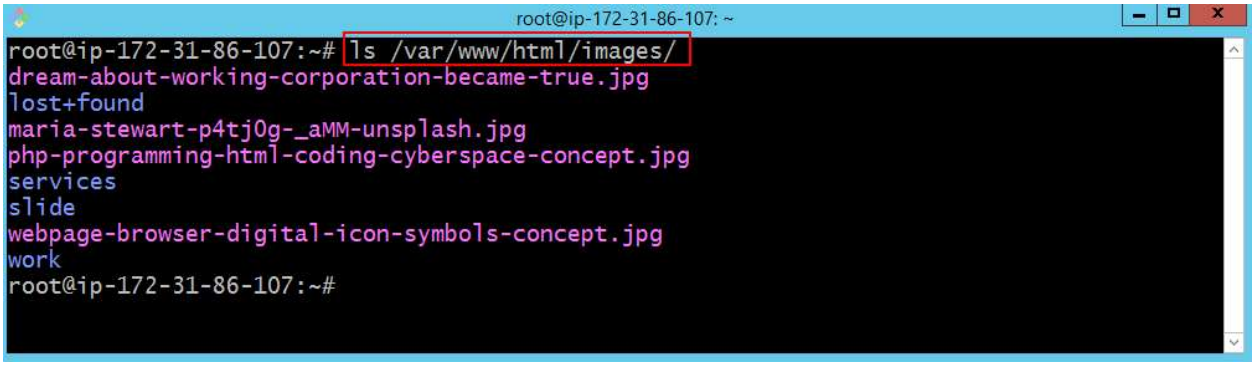
At this stage, you need to restart the **Apache** service



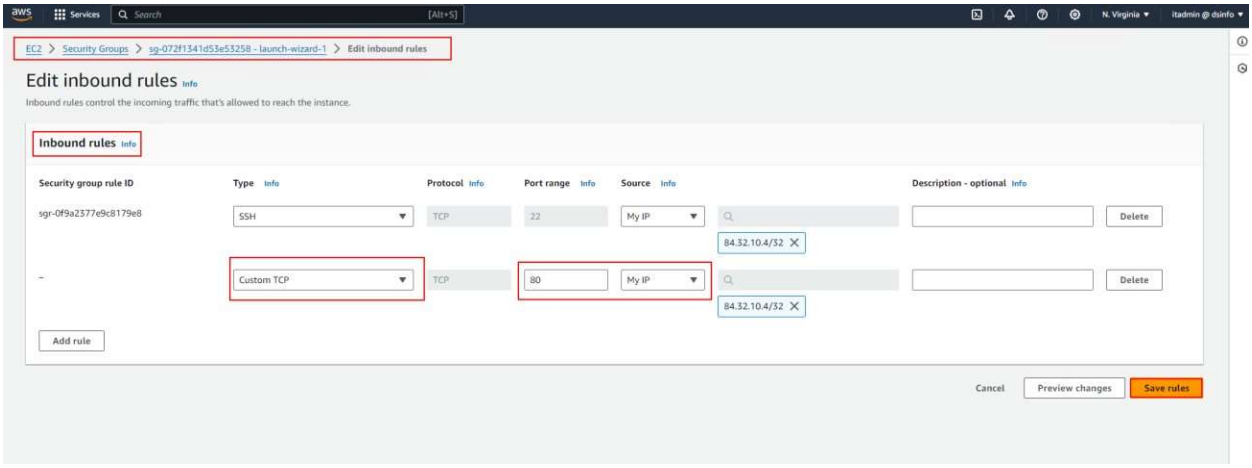
At this stage, before testing, check the status of the **Apache** service



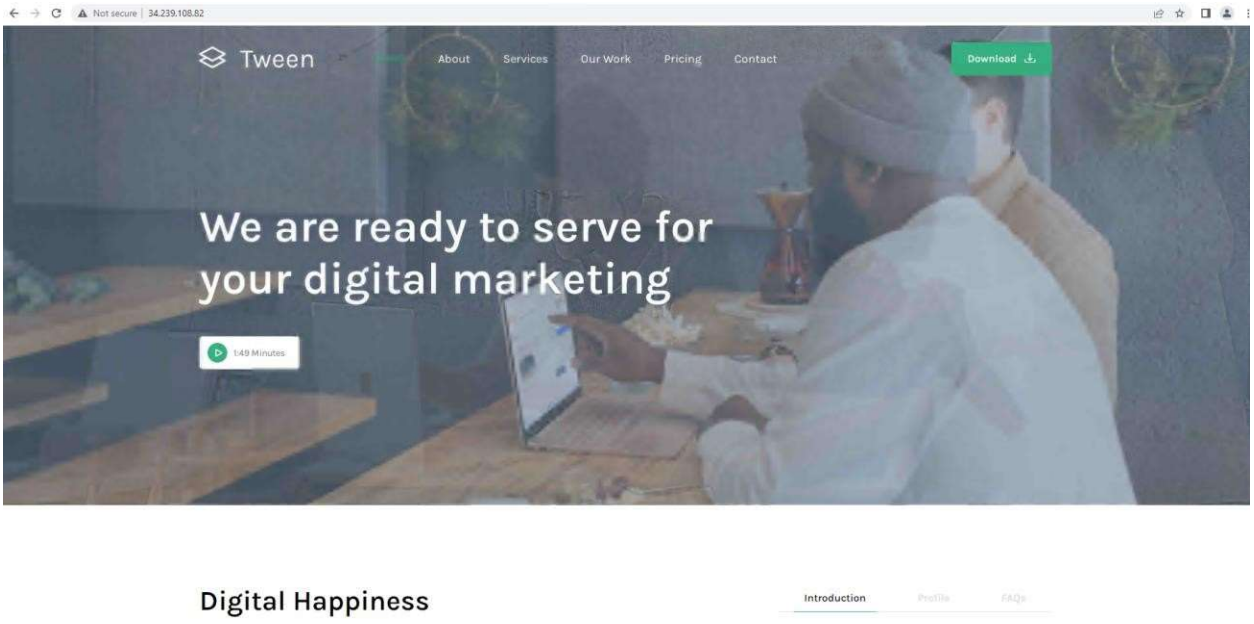
As you can see, the files are now located in the **images** folder on the web server path, and the **images** folder is currently mounted on the **new partition**.



To access the website hosted on the EC2 instance from the internet, you need to add a rule to allow **port 80** in the **Inbound Rules** of your Security Group, as shown in the image below.



For the final test, simply enter the **Public IP Address** of the web server in your browser and check whether all the images in the **images** folder (now on the new partition) are being loaded correctly from the web server path.



As shown in the image above, the **mount operation** was completed successfully, and all the images in the **images** folder have been loaded on the website.

Introduction to EBS Snapshots

The **EBS Snapshots** feature is used for **backing up** a volume and **restoring data** in case of a failure.

When a failure occurs and you need to restore data, follow these steps:

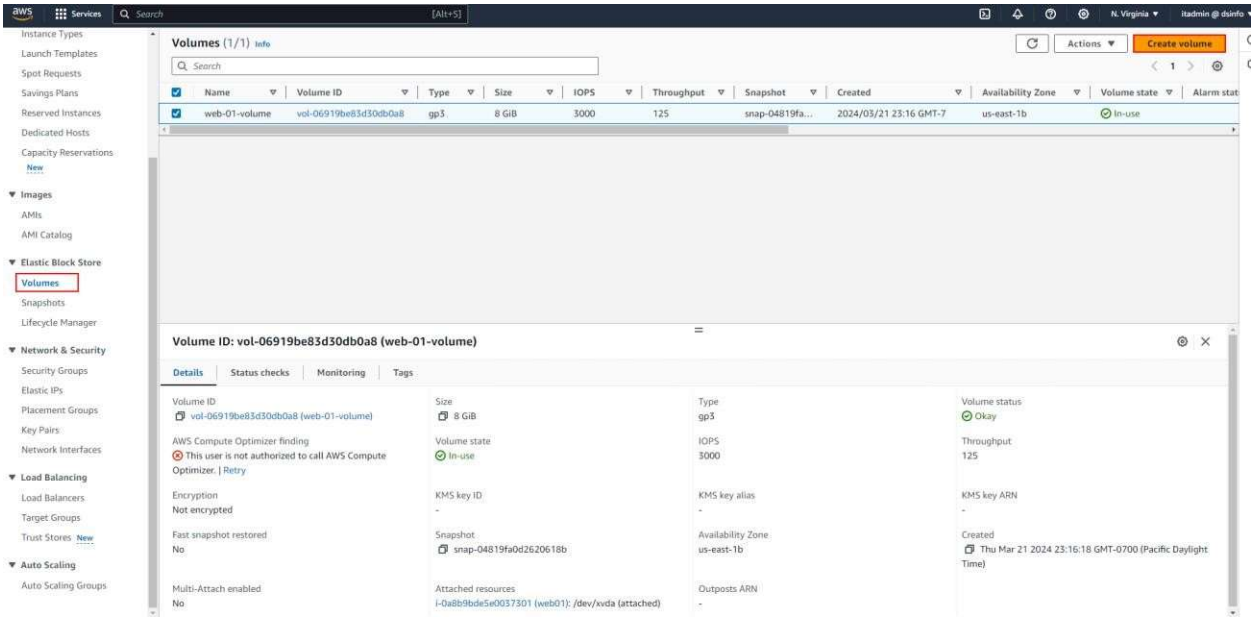
- **Unmount the Volume**.
- **Detach the Volume**.
- **Create a new volume from the snapshot**.
- **Attach the new volume** to the instance.
- **Mount the new volume**.

Therefore, the first thing that should be done is to take a **snapshot** of the volume.

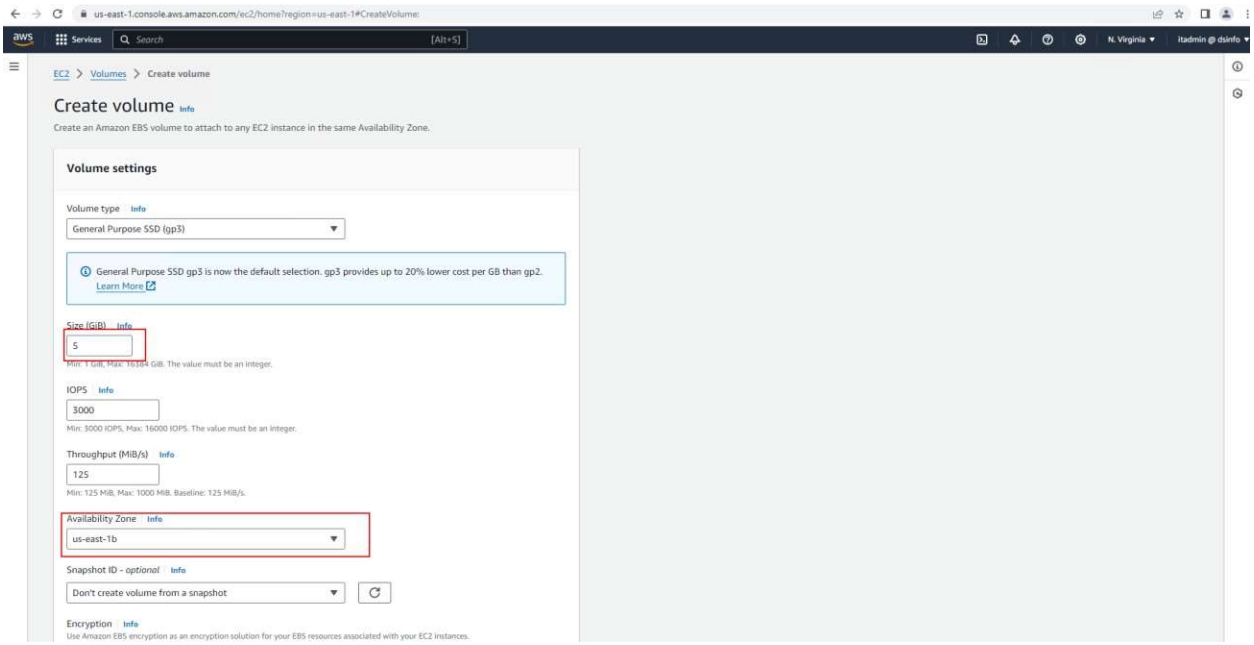
In this section, we will create a new **volume**, attach it to the **EC2 instance**, install a **database**, store the database files on the new volume, and finally, take a **snapshot** of the new volume.

How to Create a New Volume and Attach It to an EC2 Instance

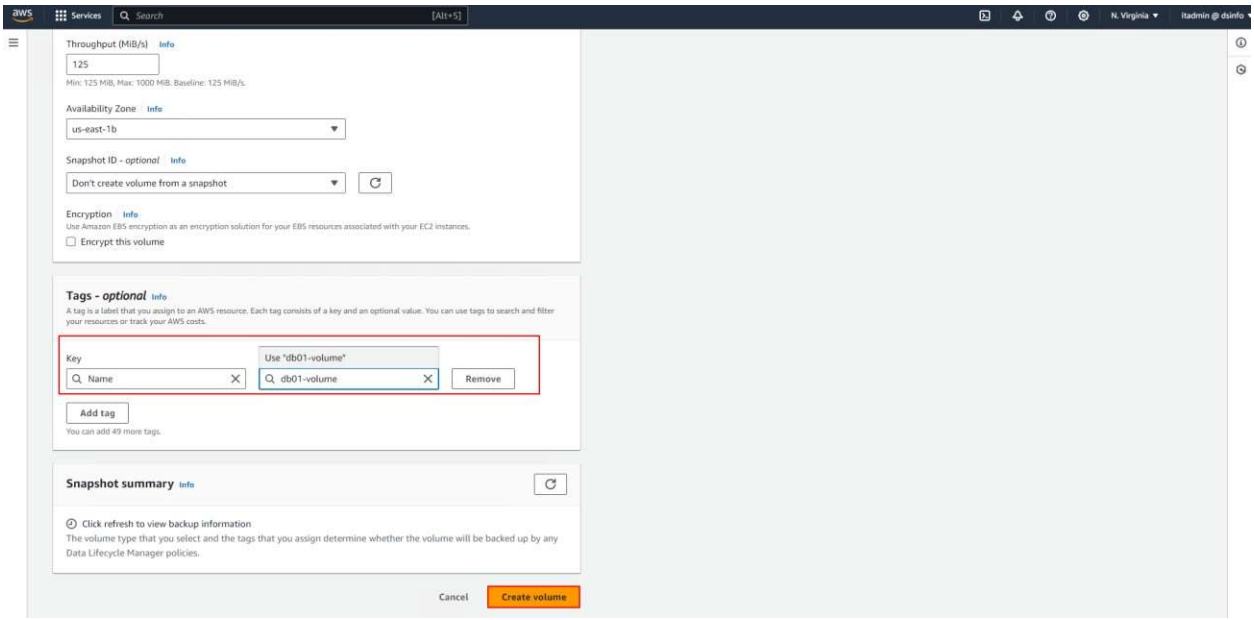
At this stage, click on the **Volumes** section, and then click the **Create Volume** button.



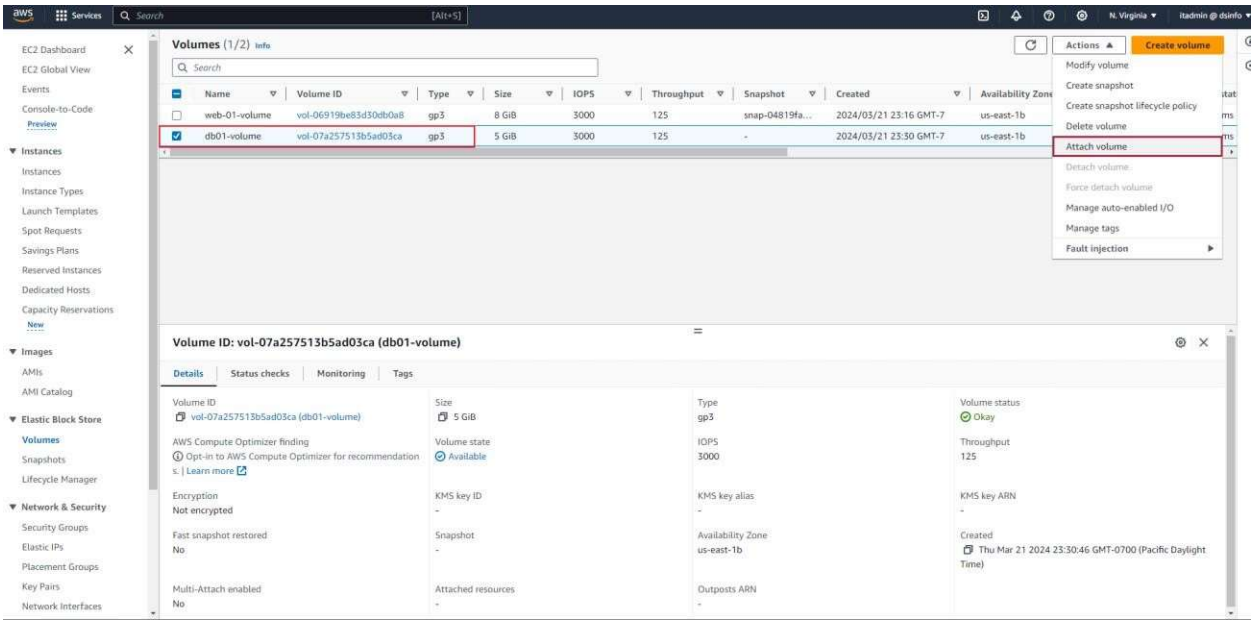
At this stage, specify the **Volume Size** and **Availability Zone**. Make sure to select the **Availability Zone** that matches your EC2 instance.



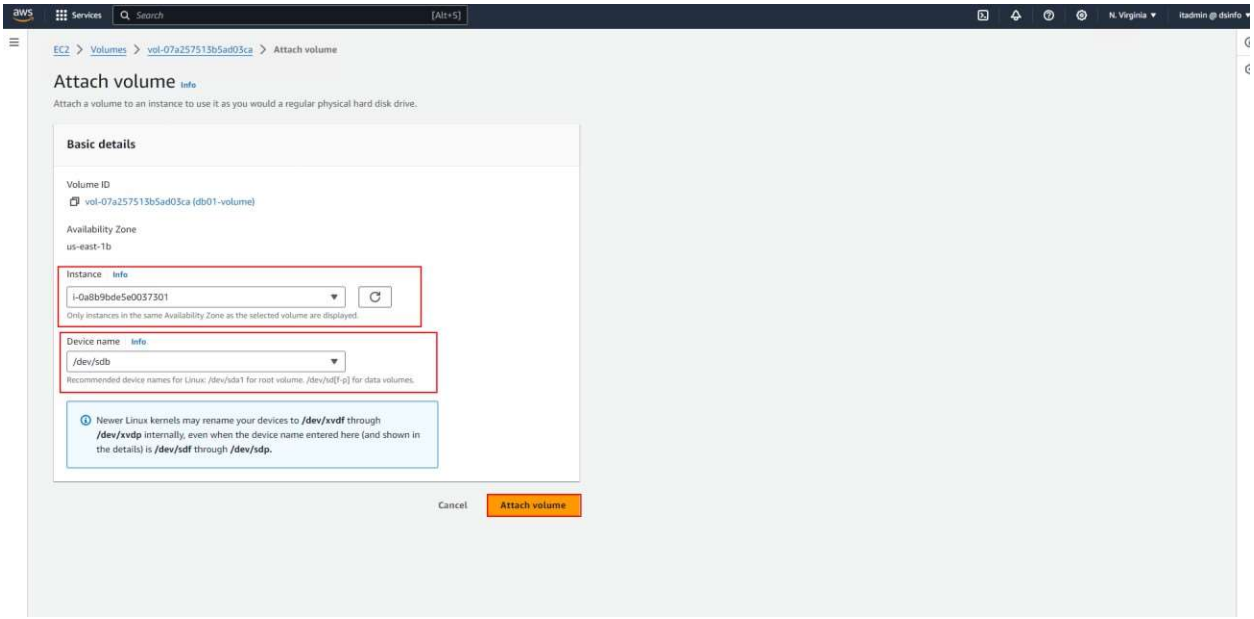
At this stage, specify a **Tag** for the volume, and then click on the **Create Volume** button to create the new volume.



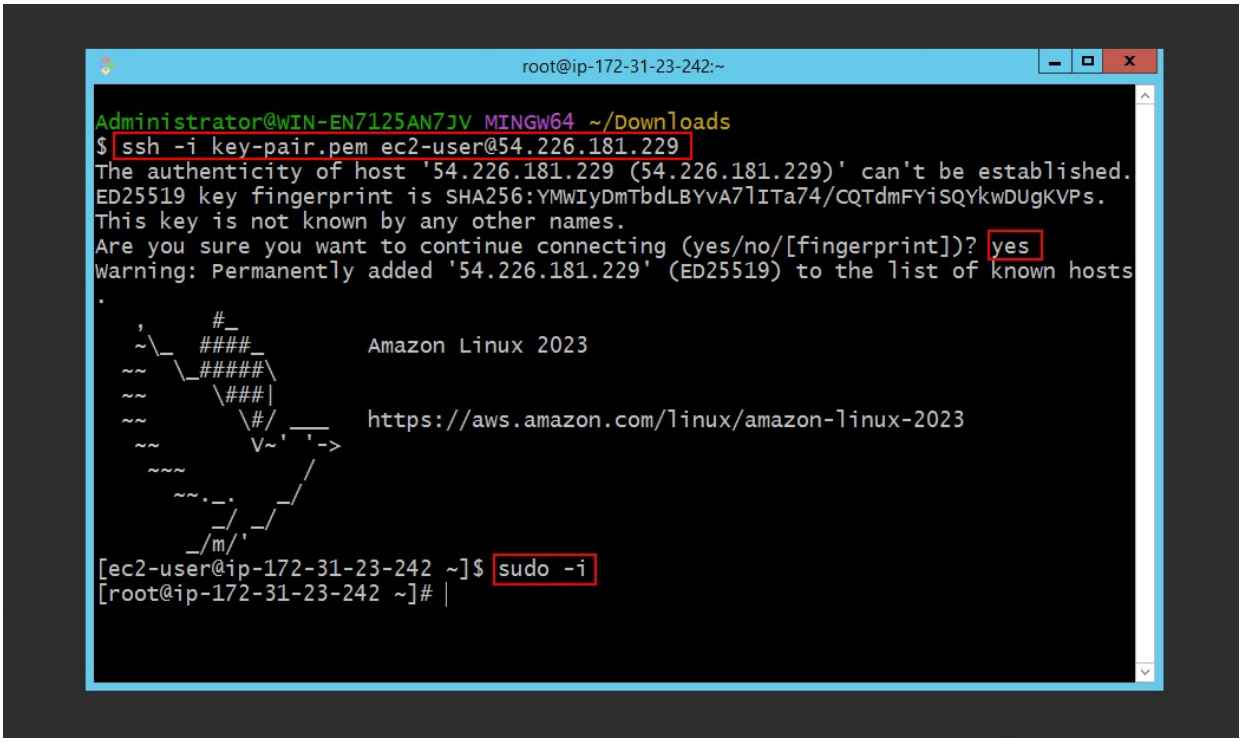
At this stage, select the **new volume**, then from the **Actions** menu, click on **Attach Volume**.



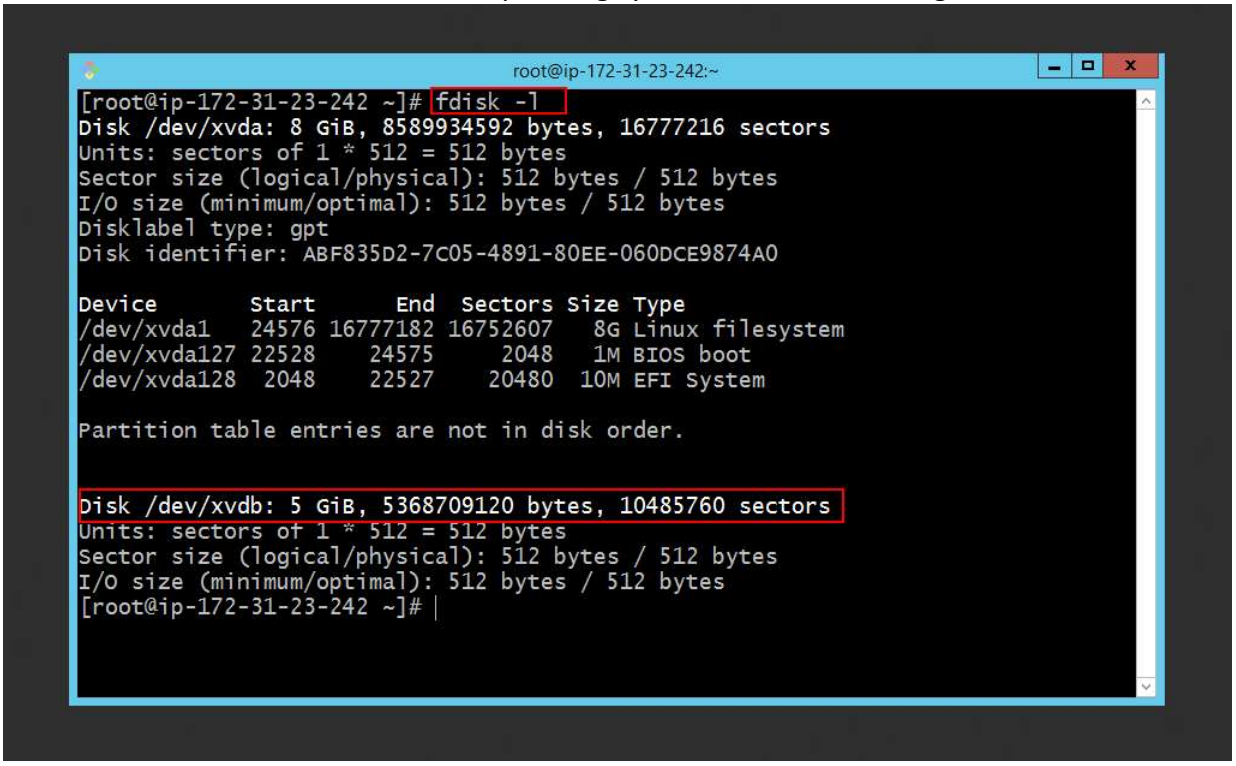
At this stage, from the **Instance** section, select the desired instance, and then click the **Attach Volume** button.



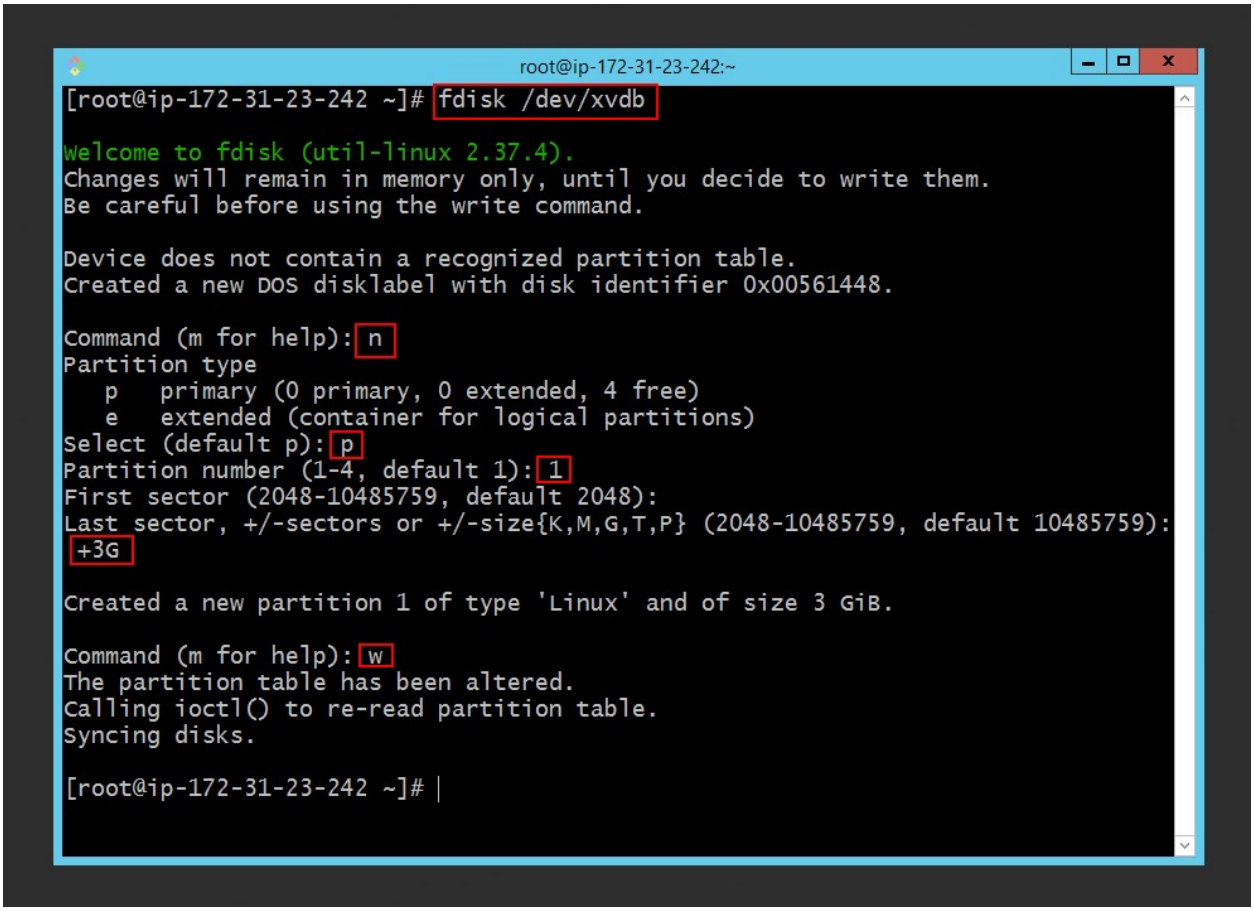
Then, SSH into your **EC2 instance** and switch to the **Root** user environment by using the following command



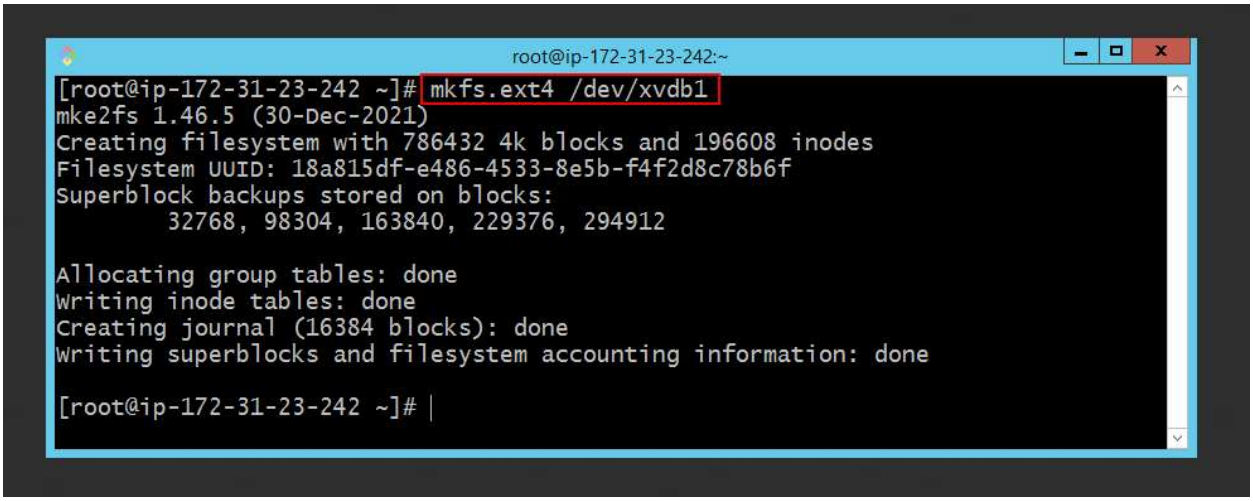
To view the new volume in the Linux operating system, use the following command



Use the following command to partition the new volume. In this command, the switch **n** stands for **New Partition**, **p** stands for **Primary**, and **1** is the partition number. Then, specify the partition size, which we have set to **3G**, and finally press **w** to save the changes.



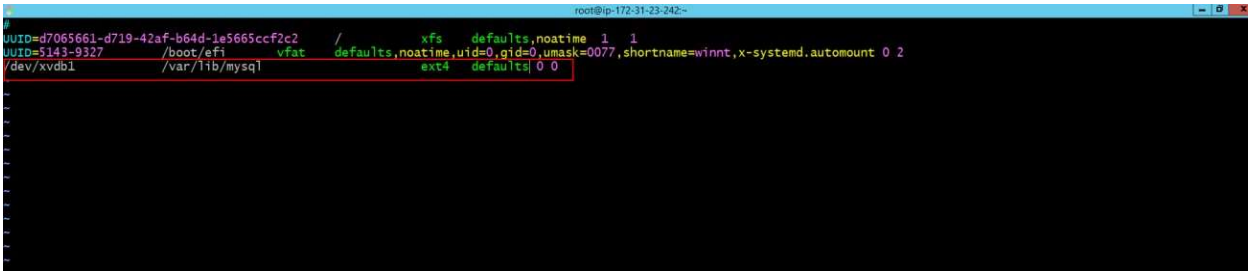
After creating the partition, you need to format it with a file system. Use the following command



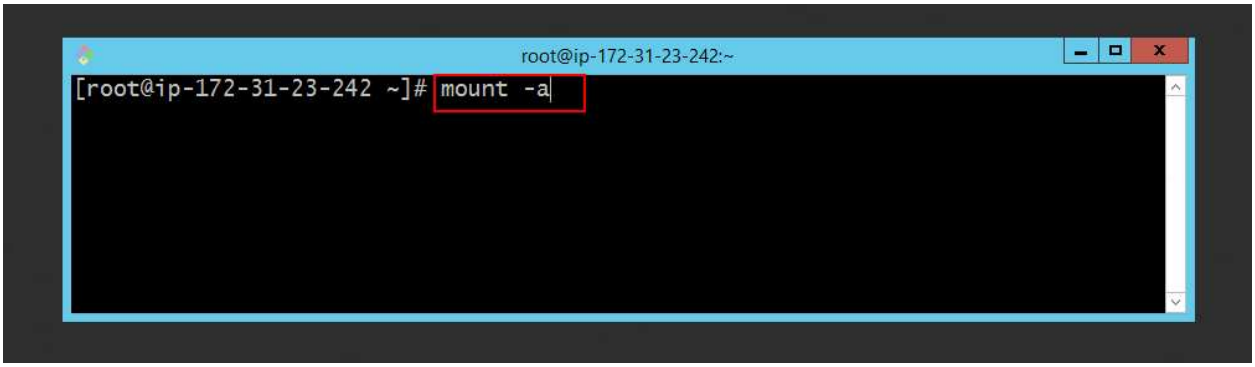
At this stage, create the directory for storing the database files using the following command

To mount the database path to the new partition, you need to edit the **fstab** file using the following command

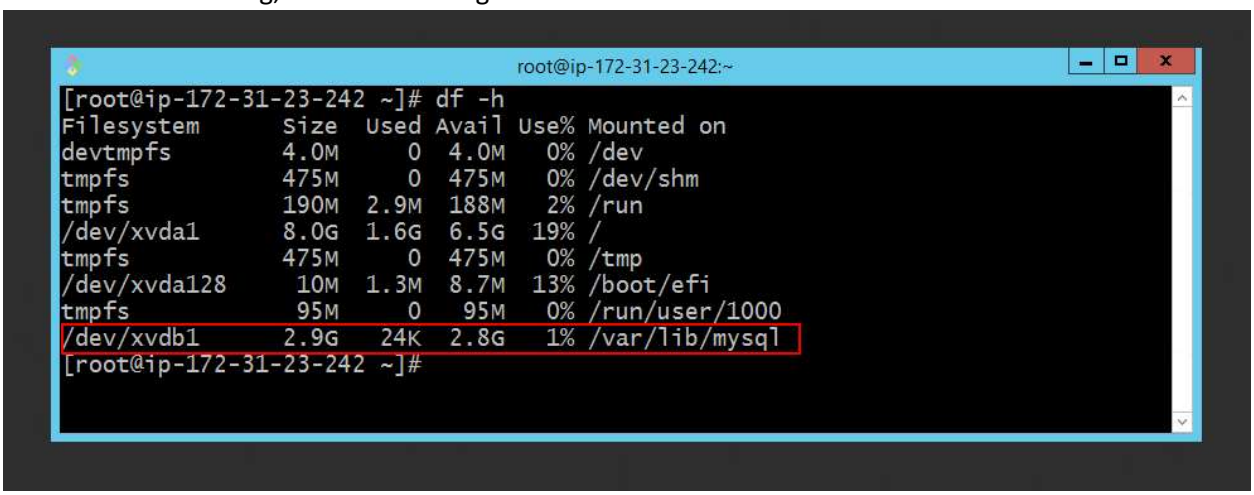
At the end of the **fstab** file, add the following line to mount the partition. After adding the line, save the file



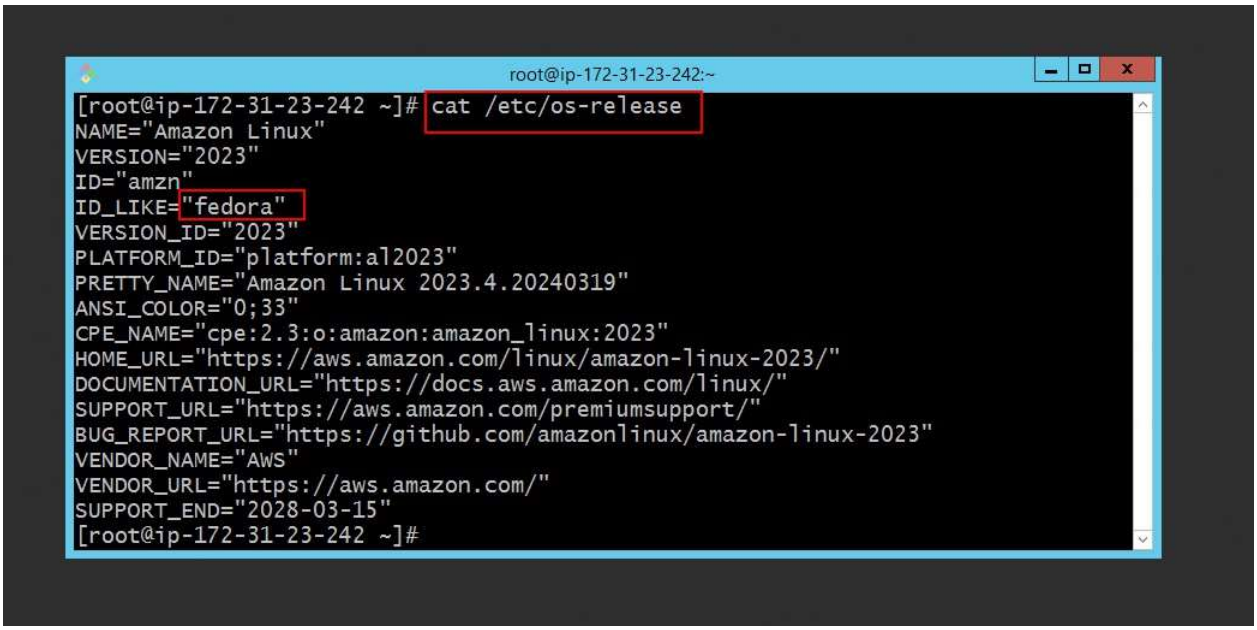
To mount the partition, use the following command



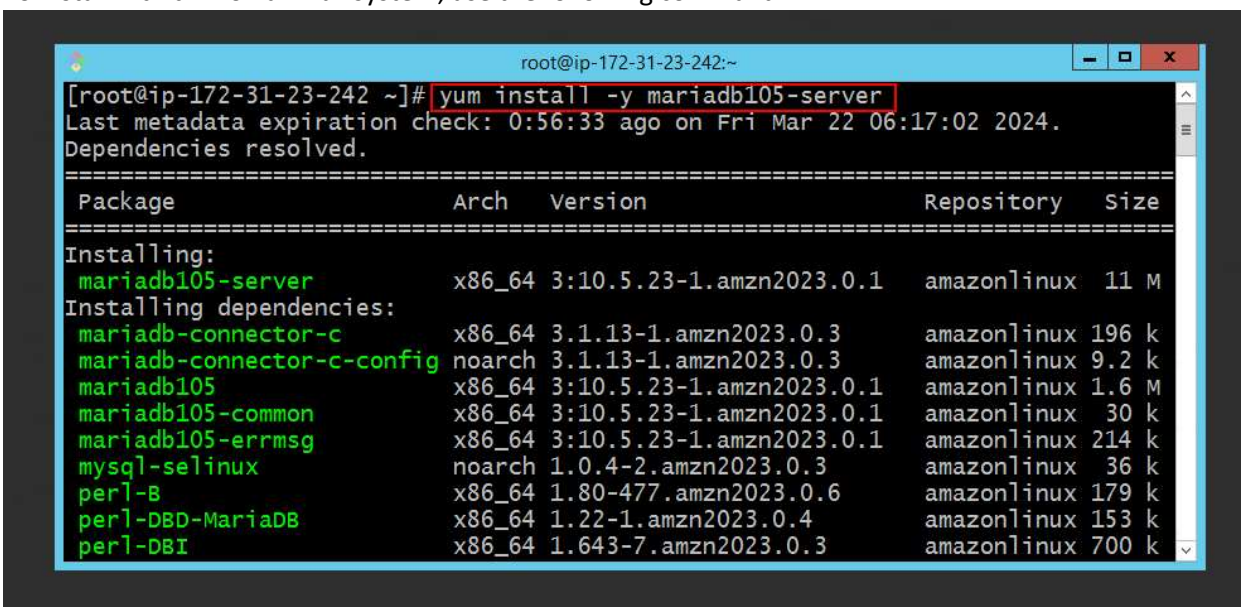
To check the mounting, use the following command



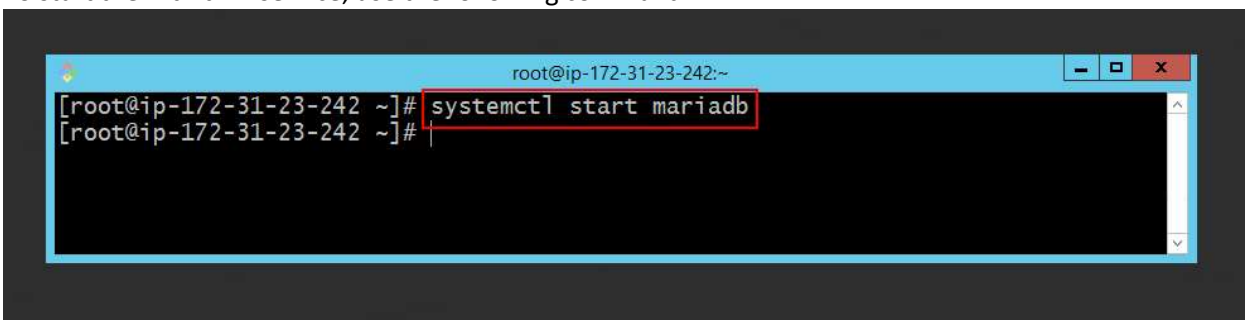
Before installing the database, check the version of the operating system using the following command



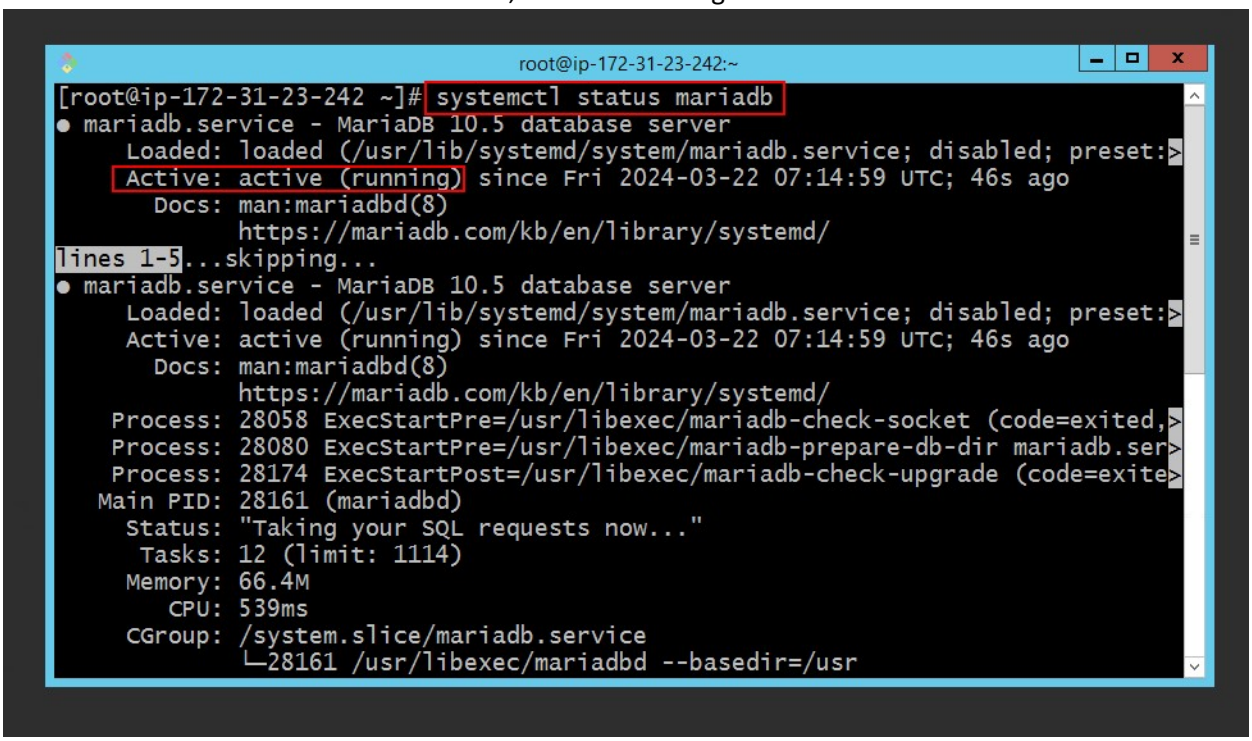
To install **MariaDB** on a Linux system, use the following command



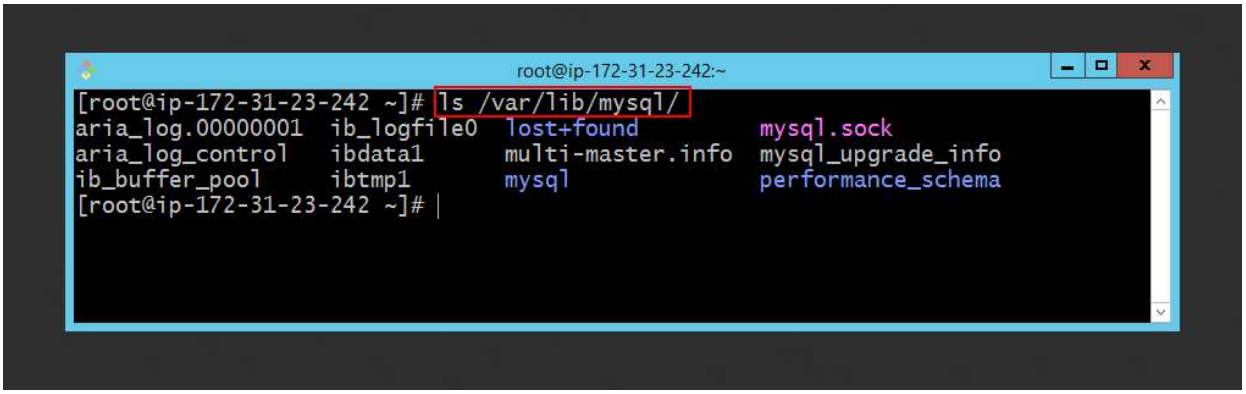
To start the **MariaDB** service, use the following command



To check the status of the **MariaDB** service, use the following command

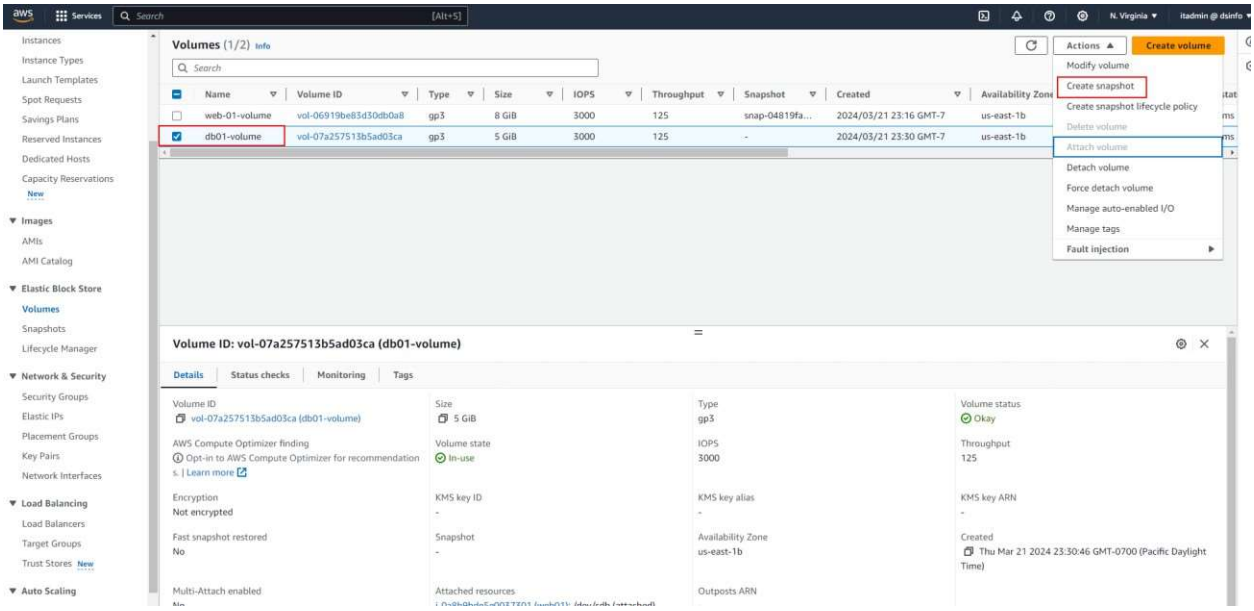


As shown in the image below, after installing the database, the database files are located in the path mounted on the new volume.

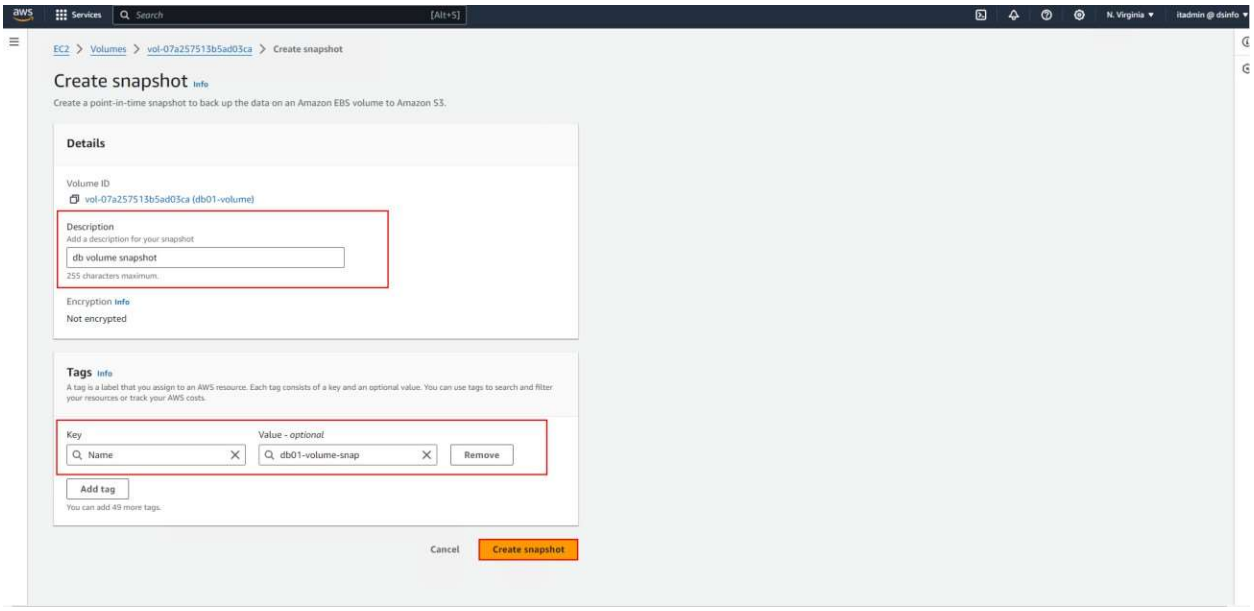


How to Take a Snapshot of a Volume

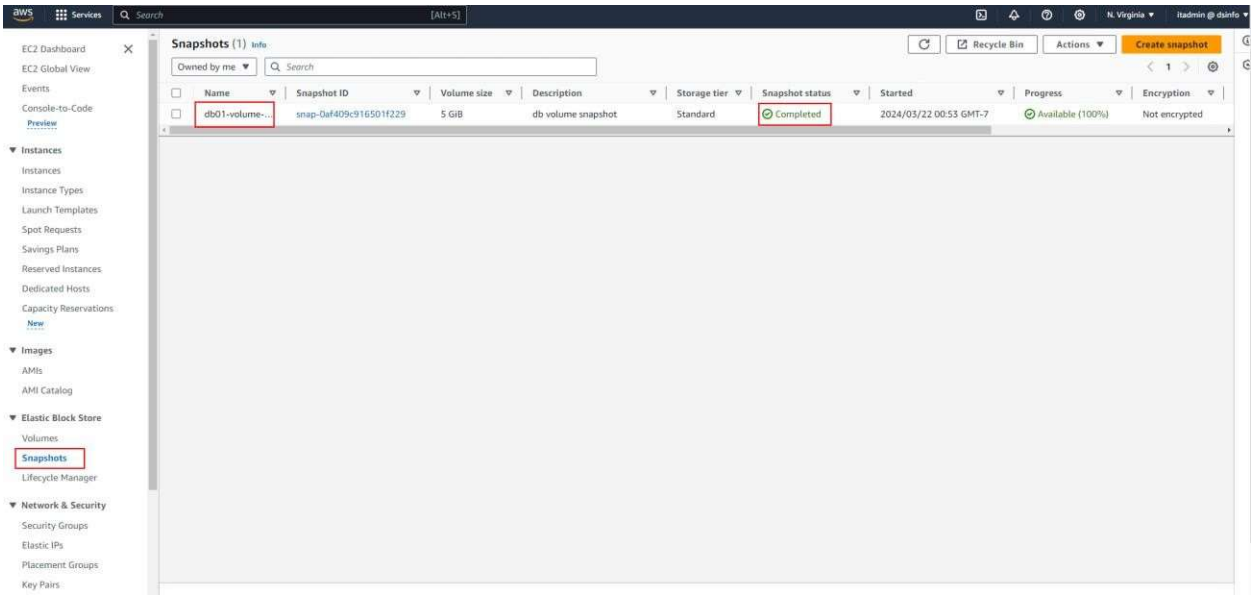
At this stage, select the **new volume**, then in the **Actions** menu, click on **Create Snapshot**.



At this stage, provide a **description** in the **Description** field, define a **tag** if needed, and finally, click on **Create Snapshot**.

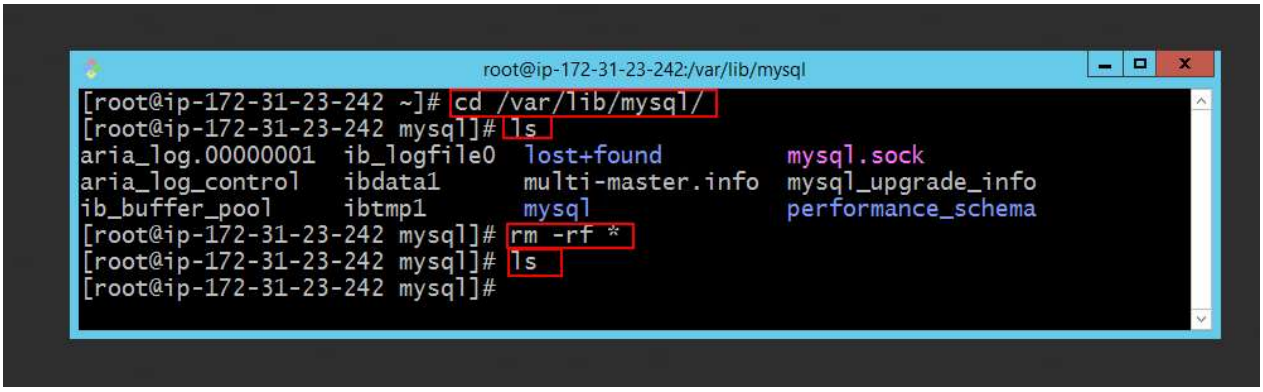


To view the created snapshot, click on the **Snapshots** section.

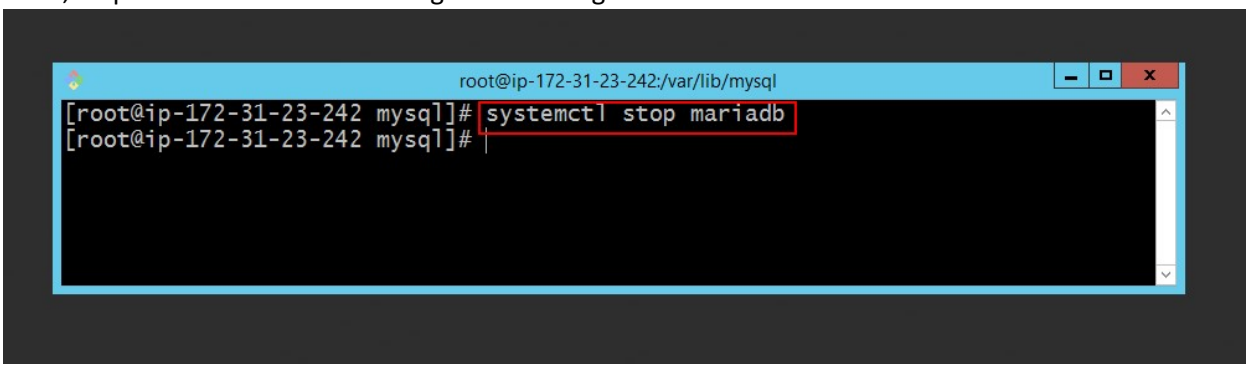


How to Use a Snapshot

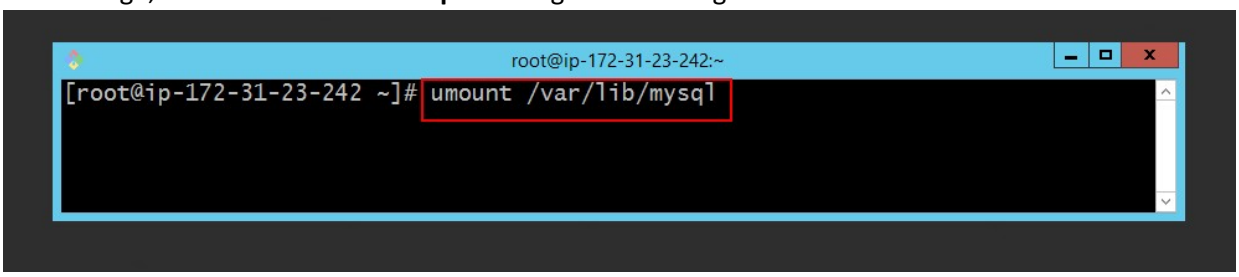
At this stage, first, delete all the files in the **database path**. Be aware that you currently have a **snapshot** of this volume, so you can restore the data



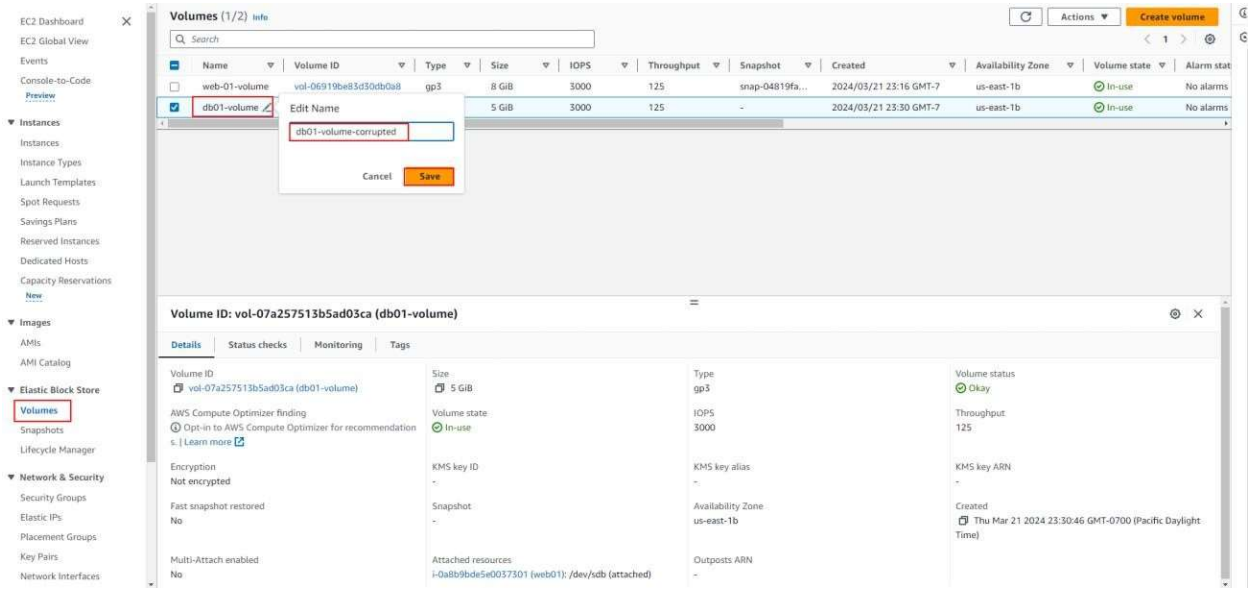
Then, stop the **MariaDB** service using the following command



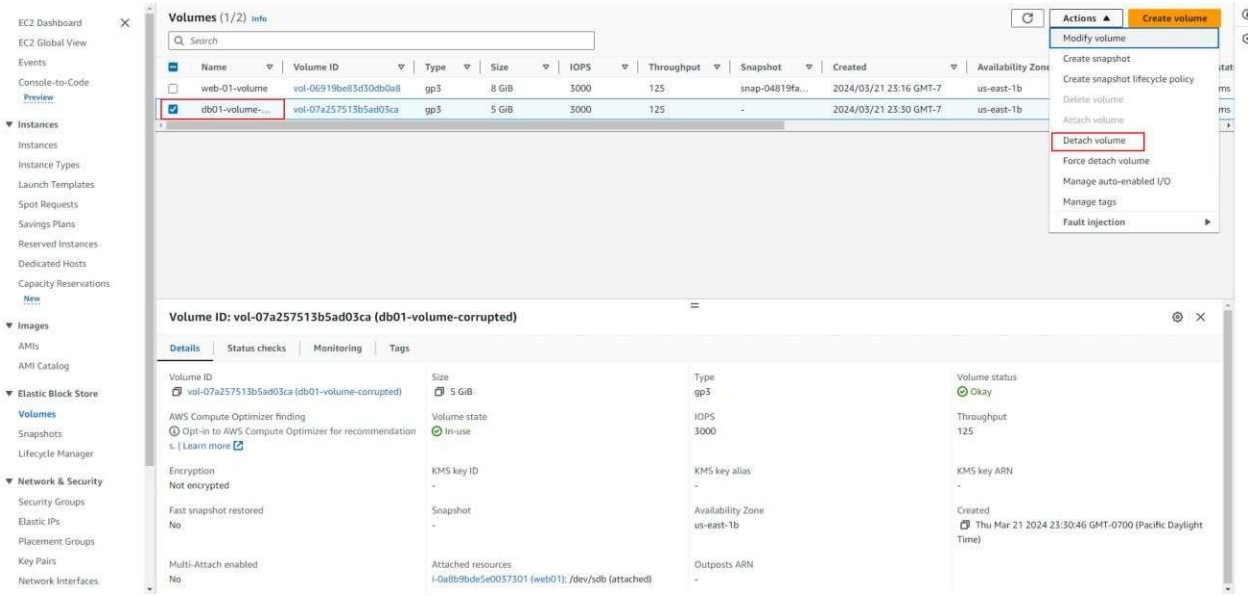
At this stage, unmount the **database path** using the following command



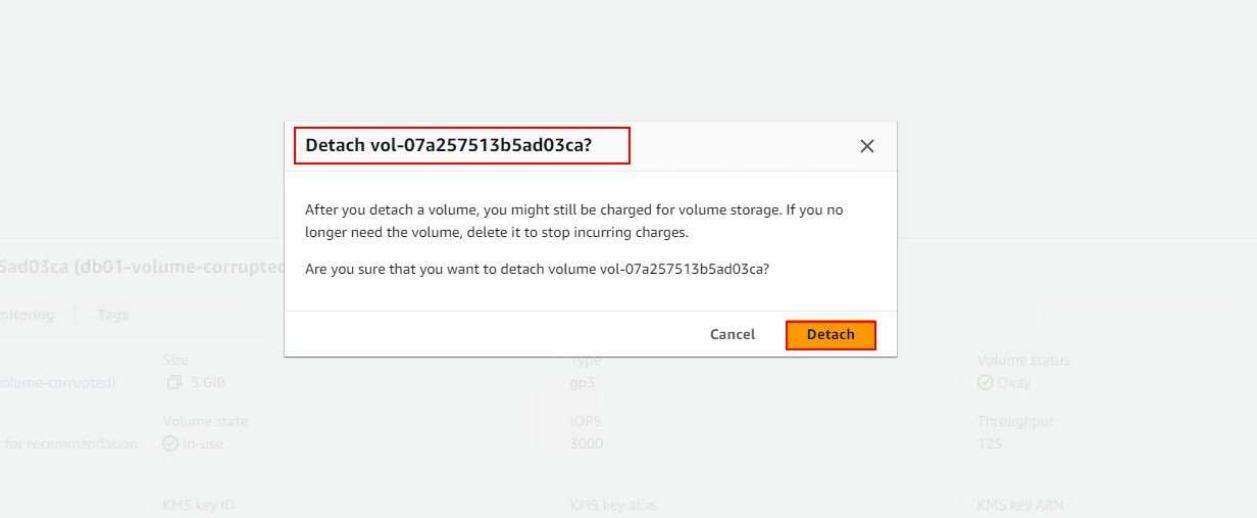
Next, click on the **Volumes** section, and then name the new volume.



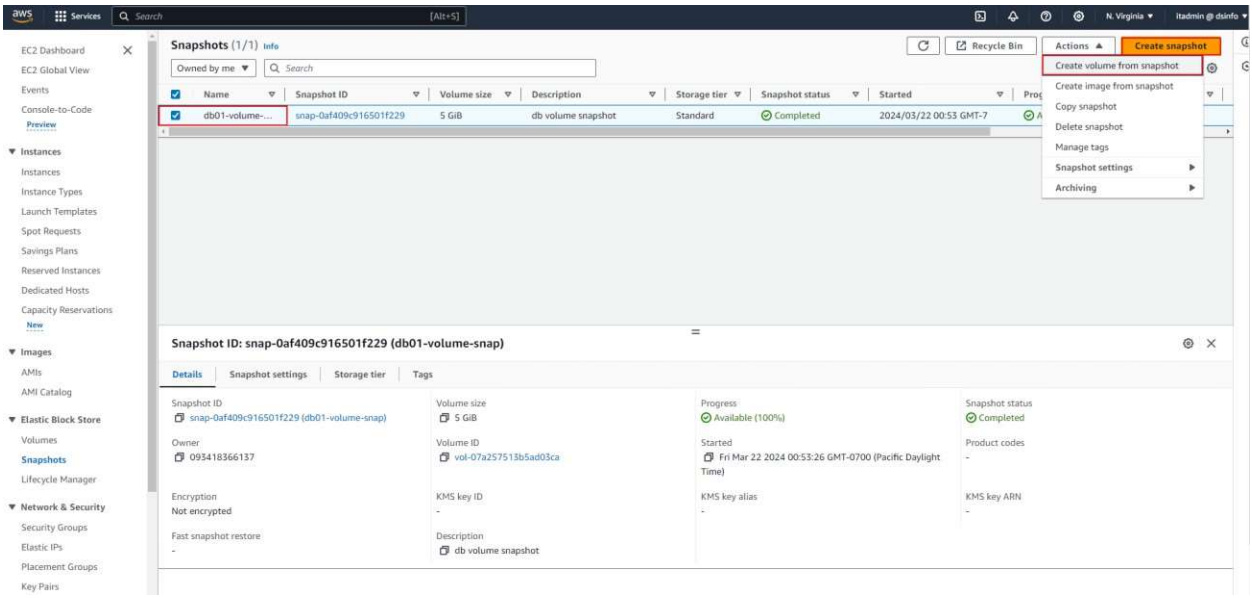
At this stage, select the **new volume**, then from the **Actions** menu, click on **Detach Volume**.



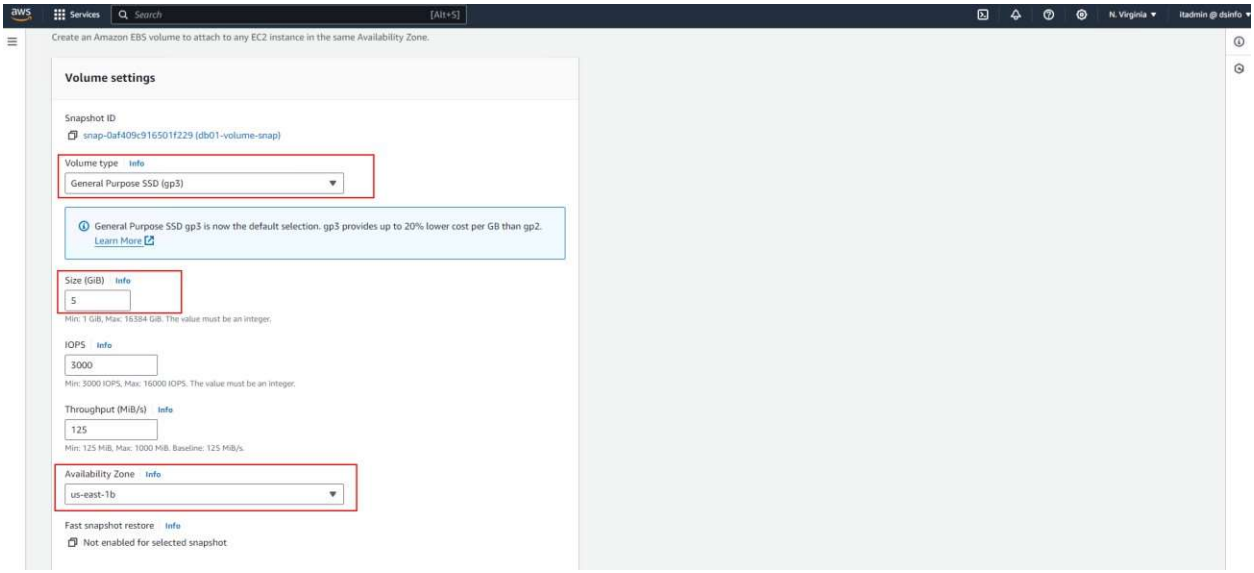
At this stage, click the **Detach** button to disconnect the volume from the EC2 instance.



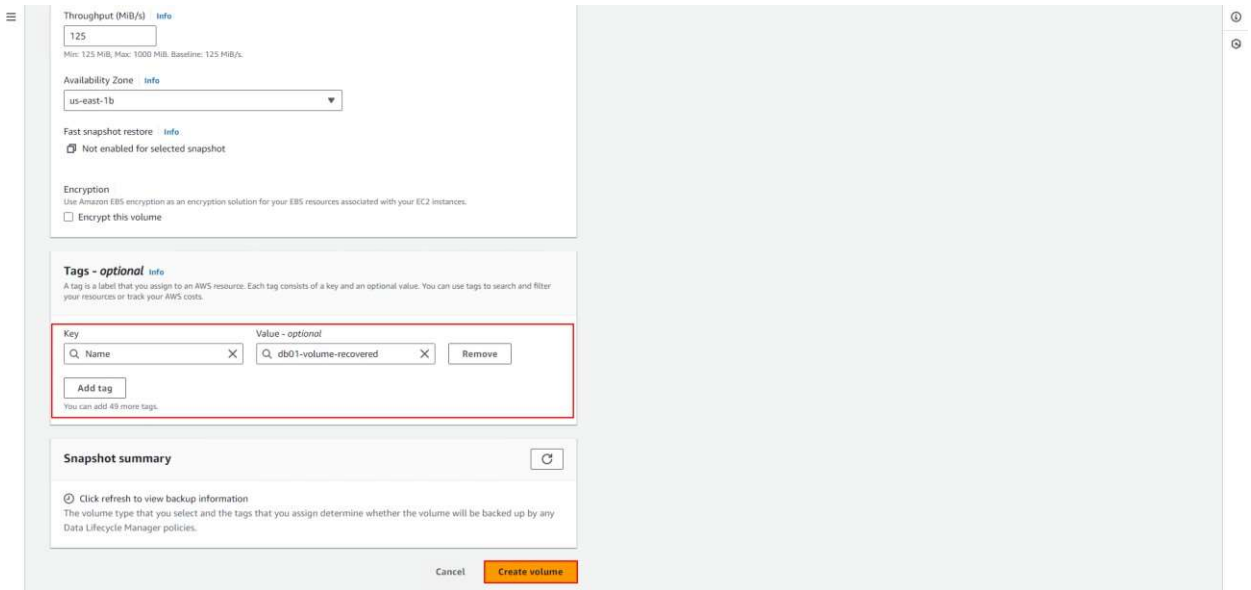
Next, in the **Snapshots** section, click on the snapshot that we took in the previous steps, and from the **Actions** menu, click on **Create Volume from Snapshot**.



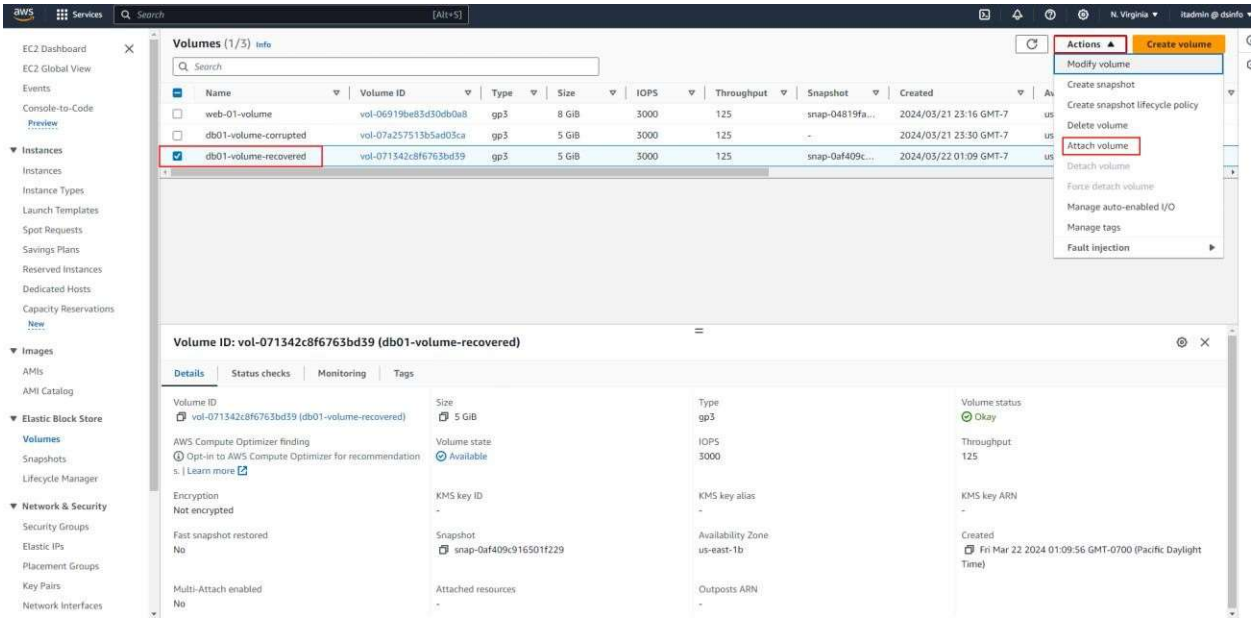
At this stage, specify the **Volume Type**, **Size**, and **Availability Zone** for the new volume. Make sure the **Availability Zone** matches the zone of your EC2 instance.



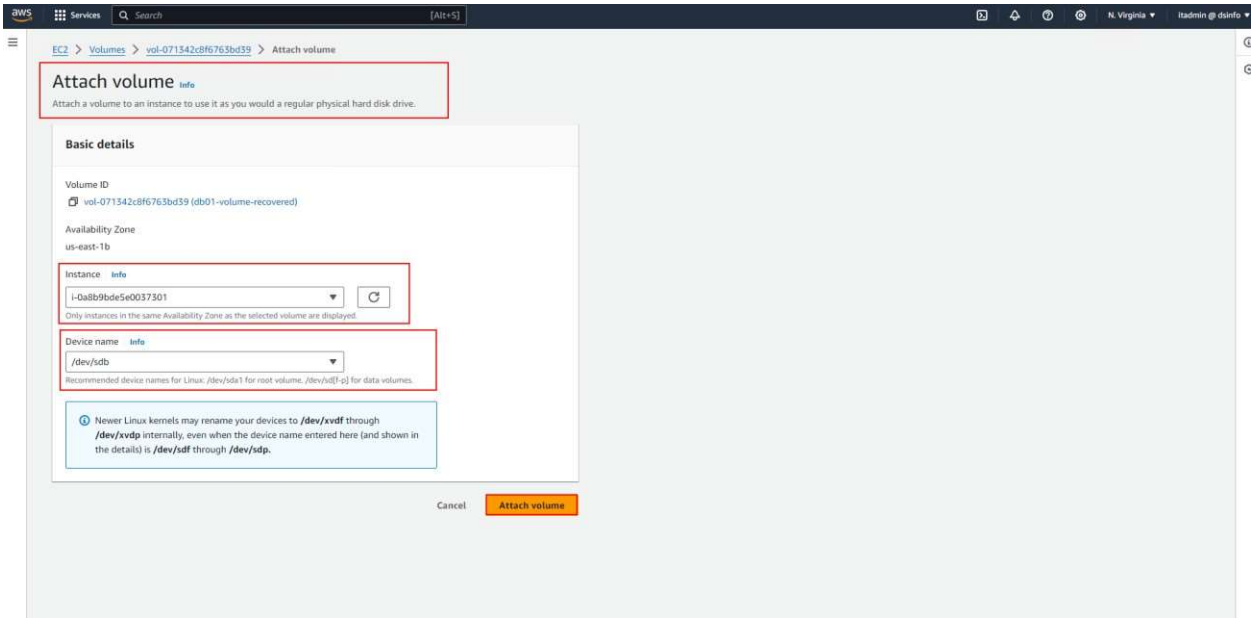
Then, define a **Tag** and click on the **Create Volume** button.



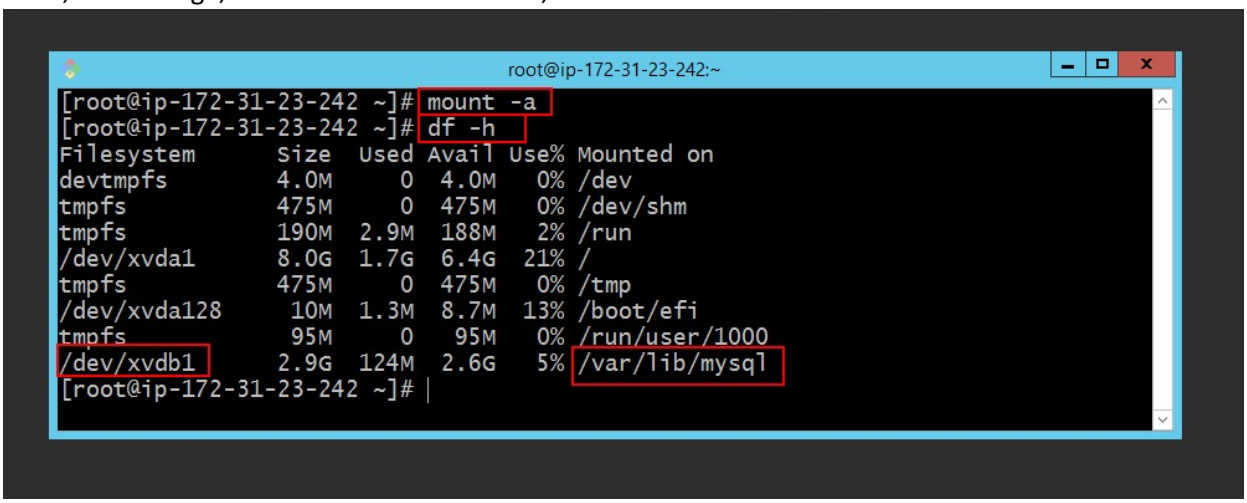
At this stage, select the **new volume**, then from the **Actions** menu, click on **Attach Volume** to attach the new volume to your EC2 instance.



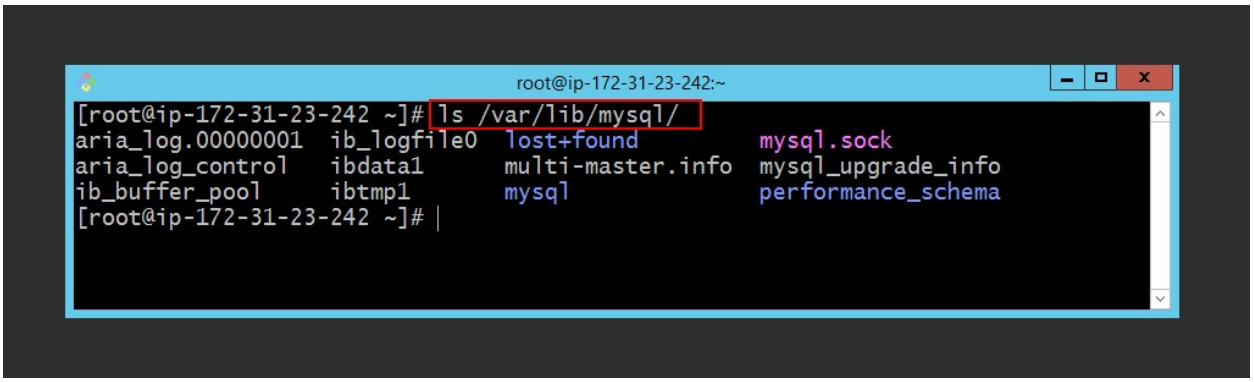
In this section, you need to select your **Instance**, and then click on **Attach Volume**.



Then, at this stage, to mount the new volume, use the command



As shown in the image below, the database files have been successfully recovered in the **database path**.



Introduction to AWS ELB

In this section, we will explore one of the exciting features of AWS Cloud, called **Elastic Load Balancer (ELB)**.

When you want to create a cluster of services, you need to set up multiple servers within the cluster. For example, let's assume you create several web servers. You need a **single endpoint** to access these web servers, and this endpoint is referred to as a **Load Balancer**.

If you're using AWS Cloud, you don't need to use your own load balancer. For example, if you're using **NGINX**, **HAproxy**, or any other load balancer, you can easily switch to using **AWS ELB**.

An **ELB** has two types of ports, known as **Frontend Port** and **Backend Port**.

The **Frontend Port** listens for user requests from the internet. For example, when you try to access **Google.com** on port **443**.

The **Backend Port** is the port associated with the service that is listening on the operating system. For example, the **Tomcat** service that runs on port **8080**.

AWS **Elastic Load Balancer** receives traffic and then distributes it across multiple **targets**.

These targets are typically **EC2 instances**, but they can also be **containers**. You can have multiple **IP addresses** across several zones. In other words, you can have a cluster across multiple zones, and the **Load Balancer** acts as the endpoint for them.

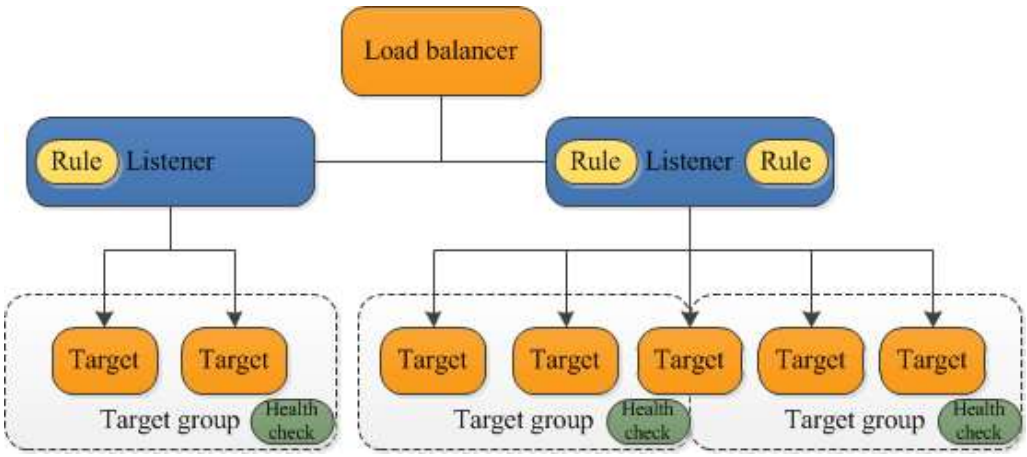
Types of AWS ELB

In AWS, there are several types of Load Balancers, including the following

Application Load Balancer

This **Load Balancer** is only for **web traffic**.

This **Load Balancer** operates at **Layer 7** of the OSI model, meaning there is no need to route traffic based on ports. It can route traffic based on the **content** of the request, such as URL paths and host headers.



Network Load Balancer

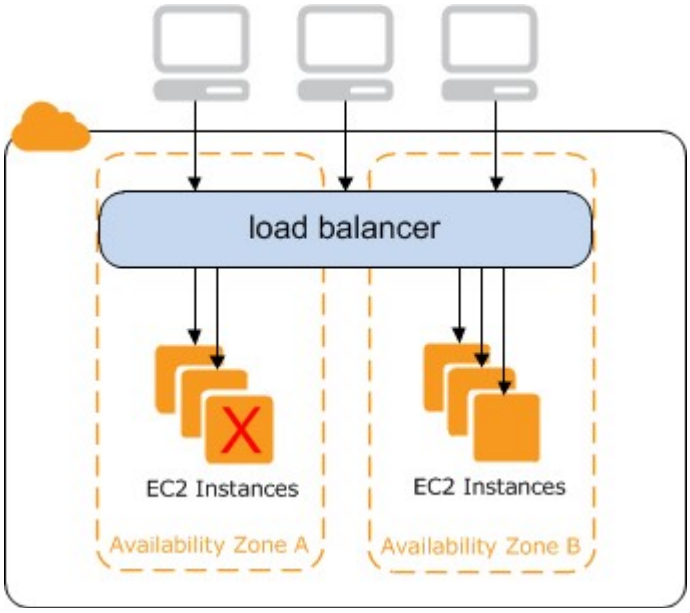
This **Load Balancer** provides **high performance** and is **very costly**.

This **Load Balancer** operates at **Layer 4** of the OSI model and can handle **millions of requests per second**.

-Classic Load Balancer

It is the simplest type of **Load Balancer** and can be easily used by anyone.

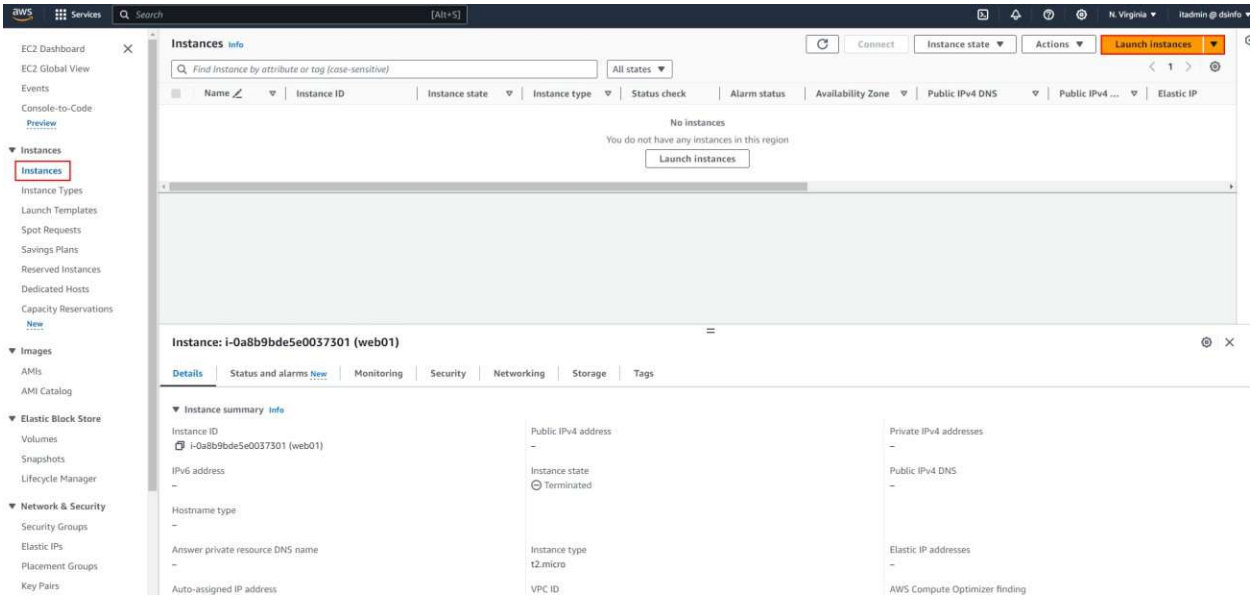
This **Load Balancer** receives incoming traffic and forwards it to the **backend servers**, operating at the **network layer** (Layer 4) of the OSI model.


load balancer
EC2 Instances
Availability Zone A
EC2 Instances
Availability Zone B

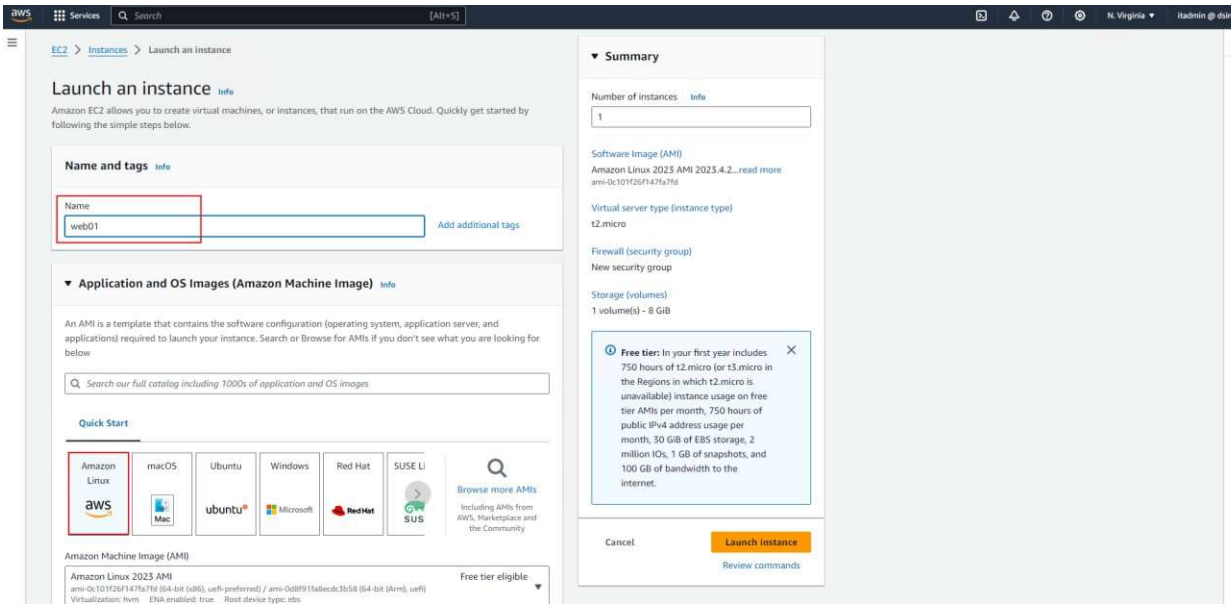
How to Set Up AWS ELB

At this stage, we intend to set up **multiple web servers** and use **HTML templates** from the Tooplate website on each server. Then, we will use a **Load Balancer** to **balance traffic** between these web servers.

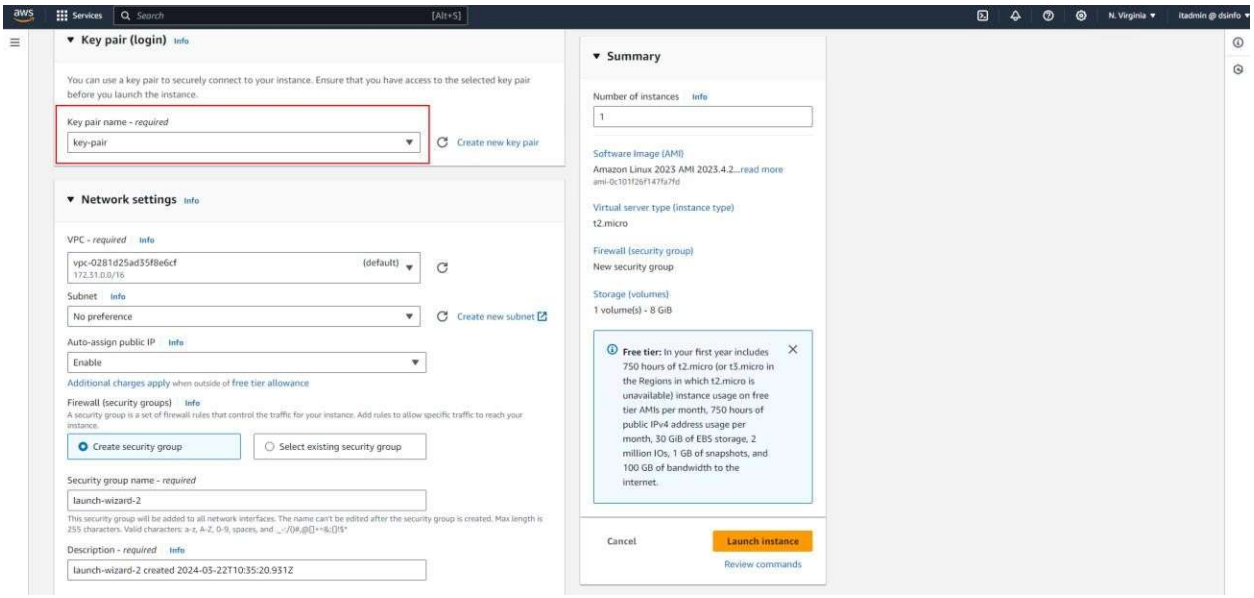
In the first step, we create an **EC2 instance** to set up a **web server**.



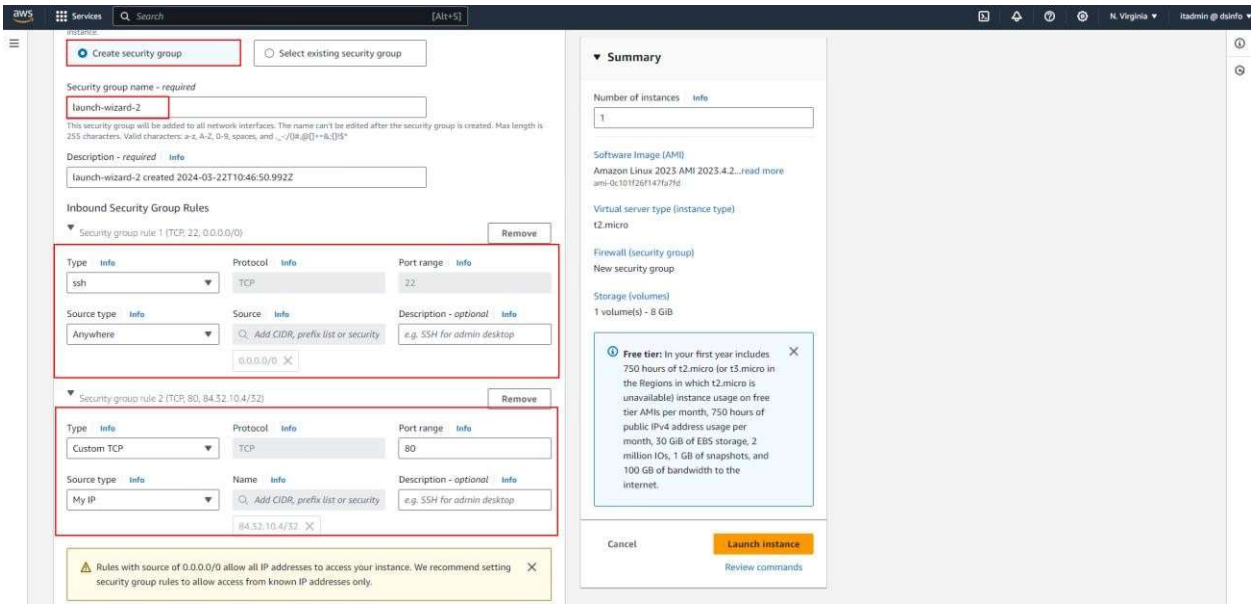
At this stage, specify a **name** for the EC2 instance and select the **AMI type**.



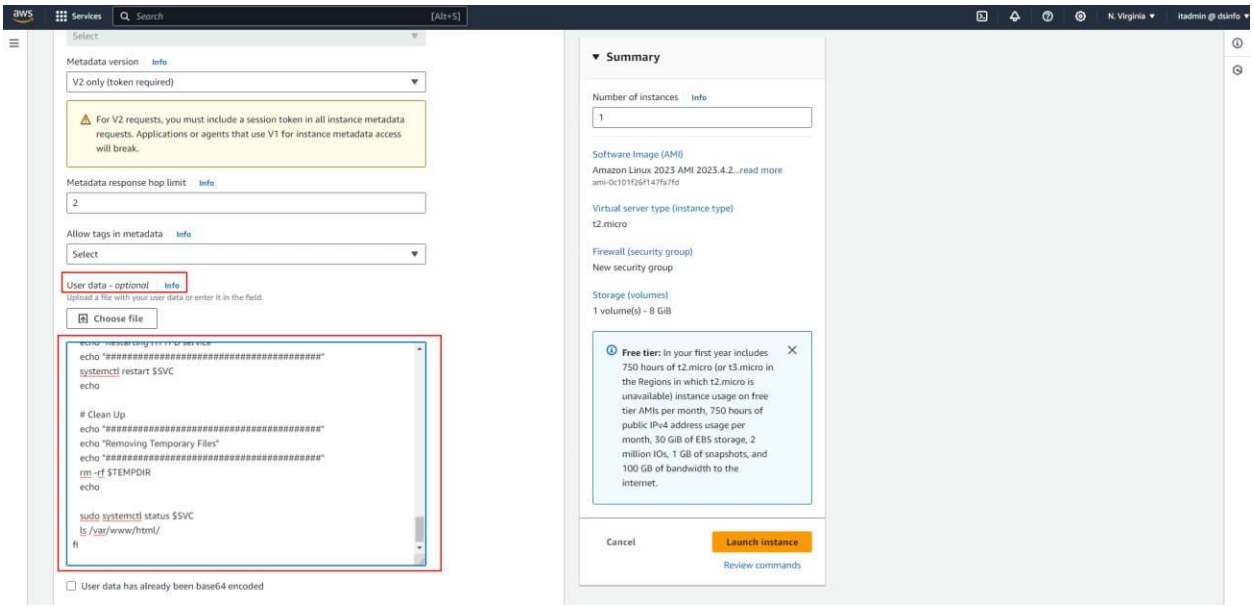
At this stage, you need to define a **Key Pair**.



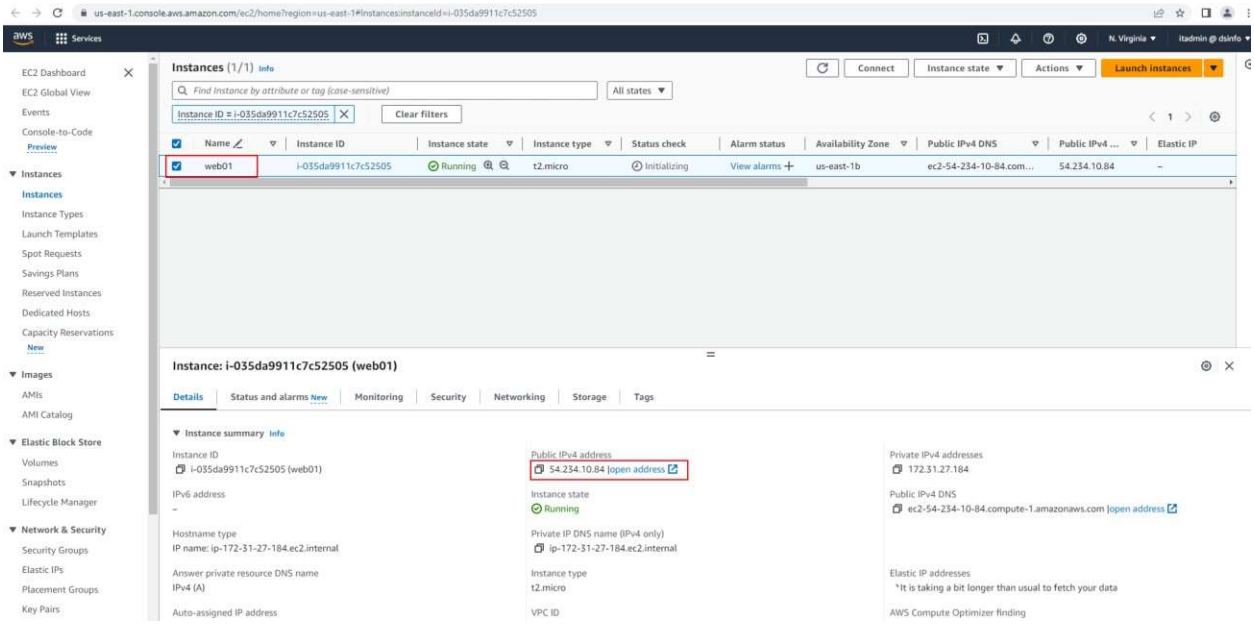
At this stage, you need to configure the **Security Group** settings for this EC2 instance. Open **ports 80** and **22** in the **Inbound Rules** section.



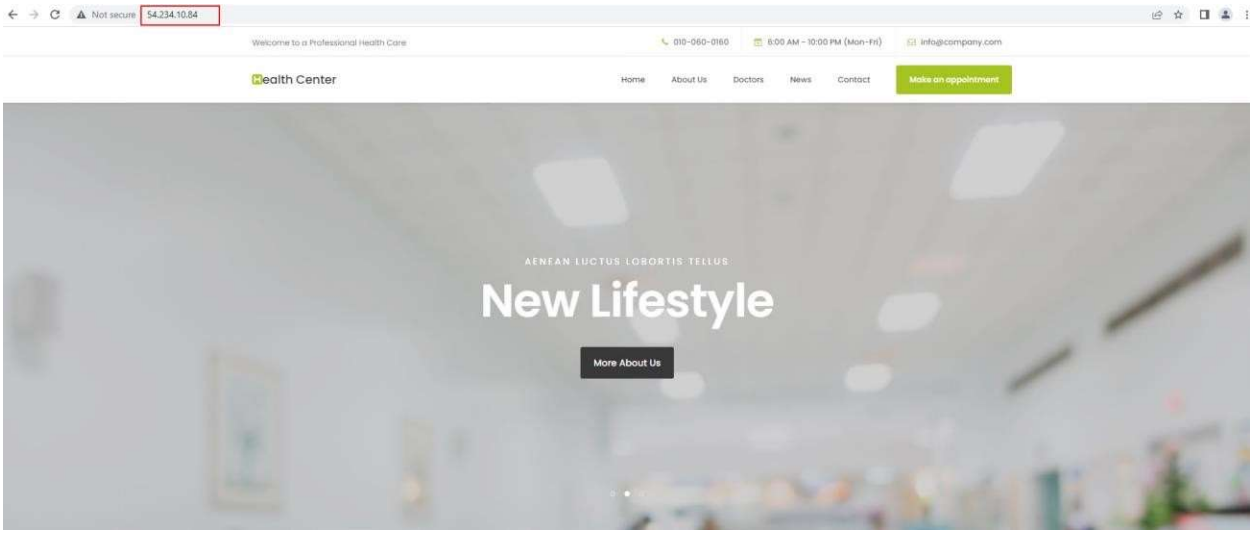
At this stage, paste the **web server setup script** into the **User Data** section.



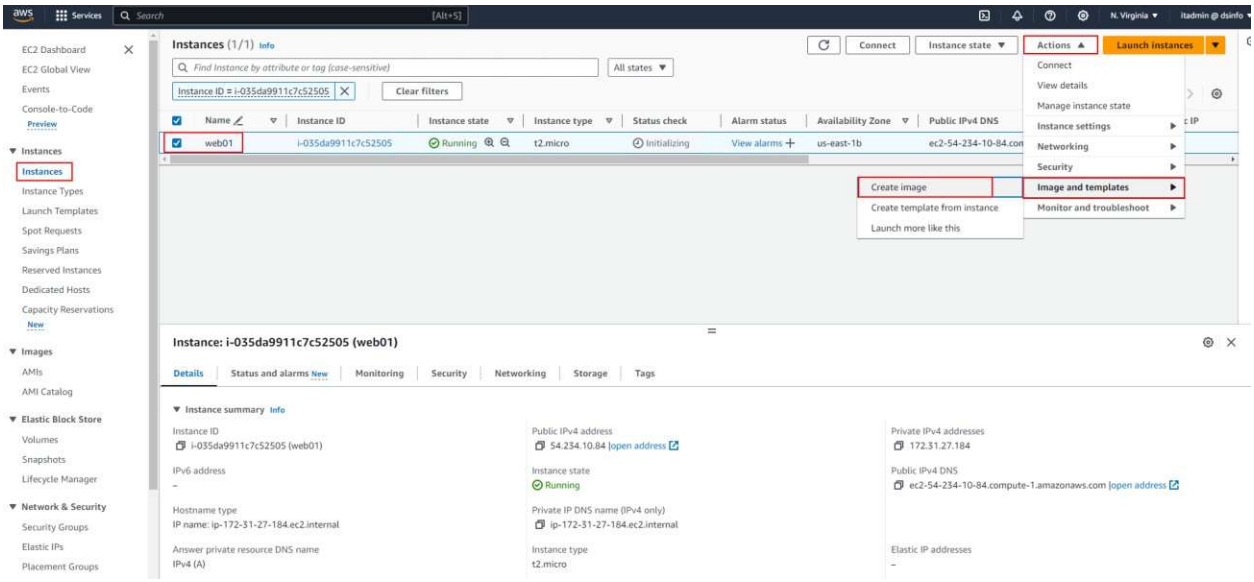
After the **EC2 instance** is created, select it and then copy its **Public IP**.



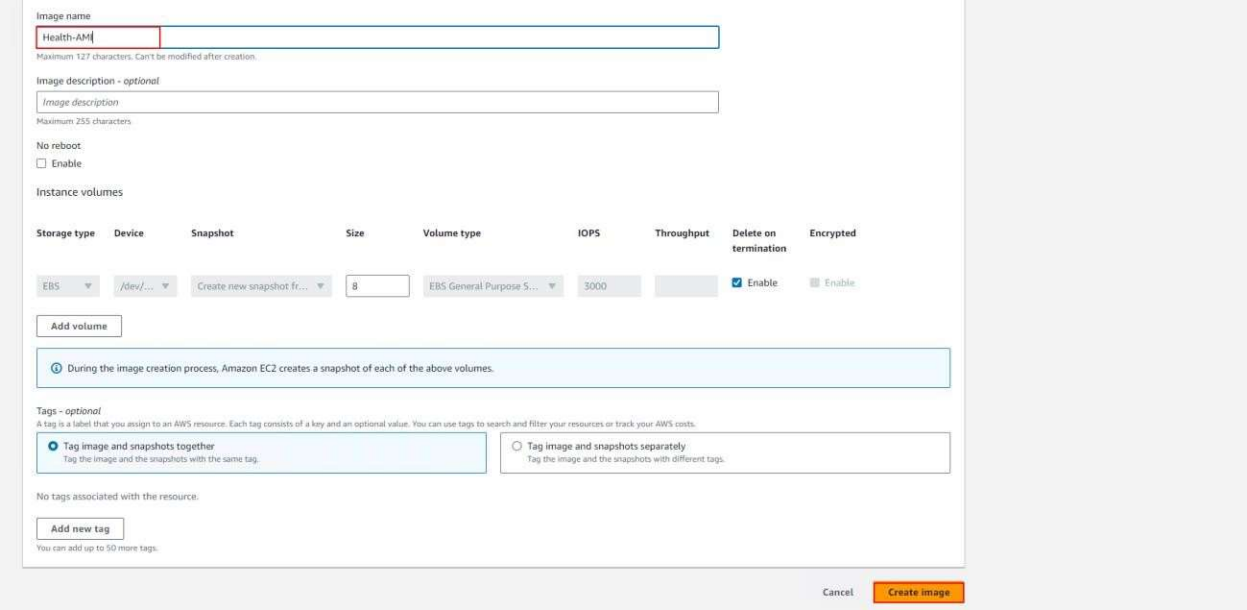
To test the web service, enter its **Public IP** in a browser. As you can see, the **web service is functioning correctly**.



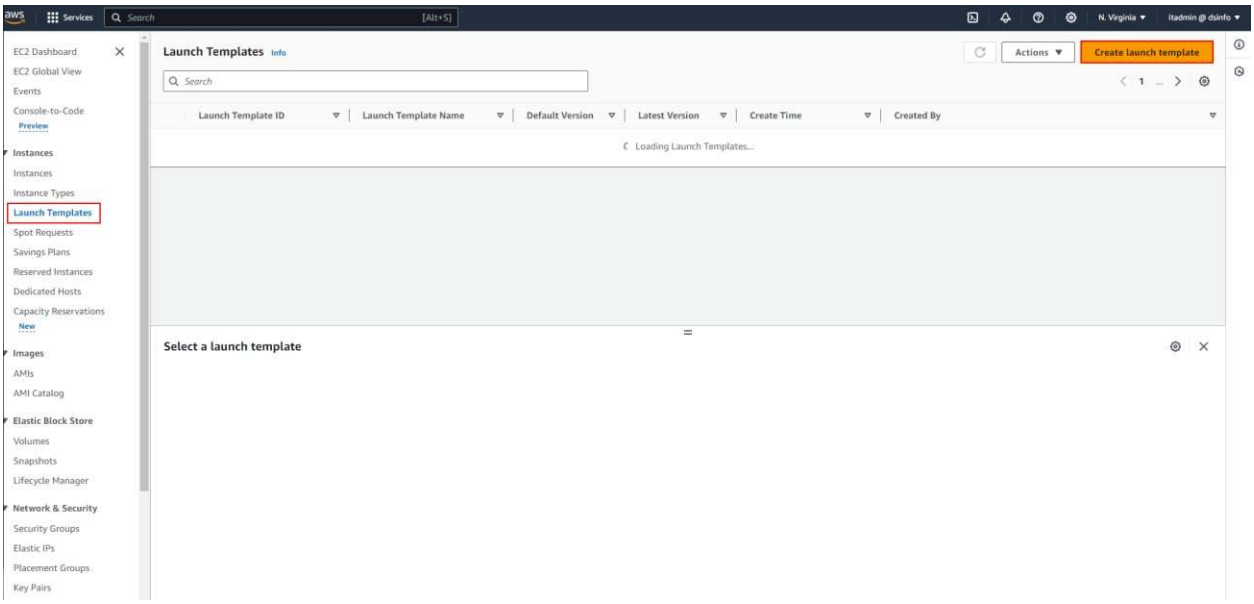
At this stage, select the **EC2 instance**, then from the **Actions** menu choose **Image and Templates**, and finally click on **Create Image**.



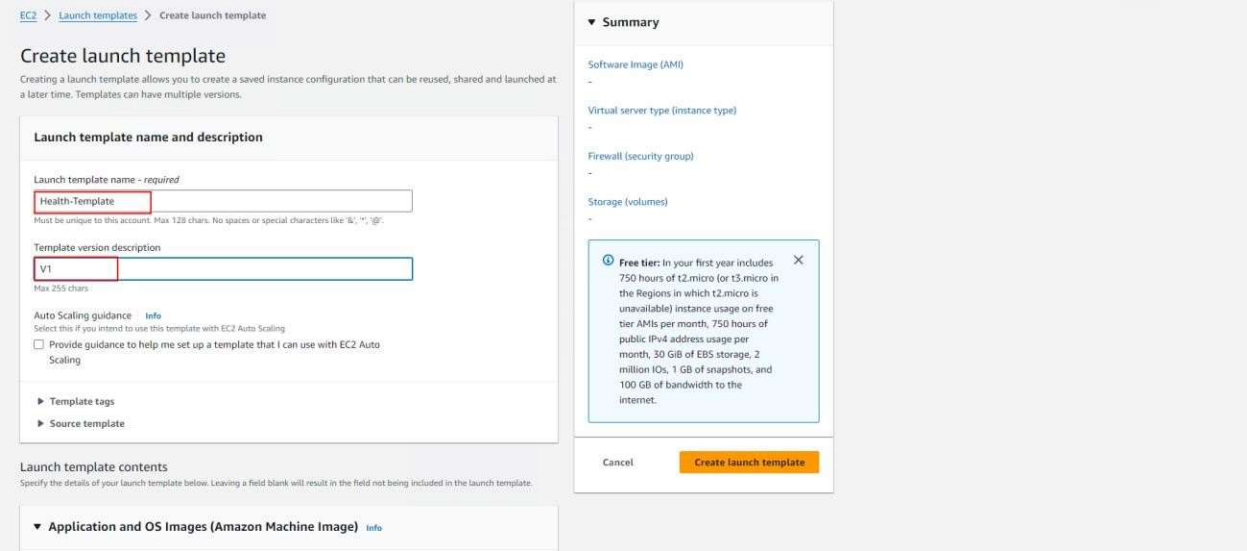
Choose a **name** for the image, then click on the **Create Image** button.



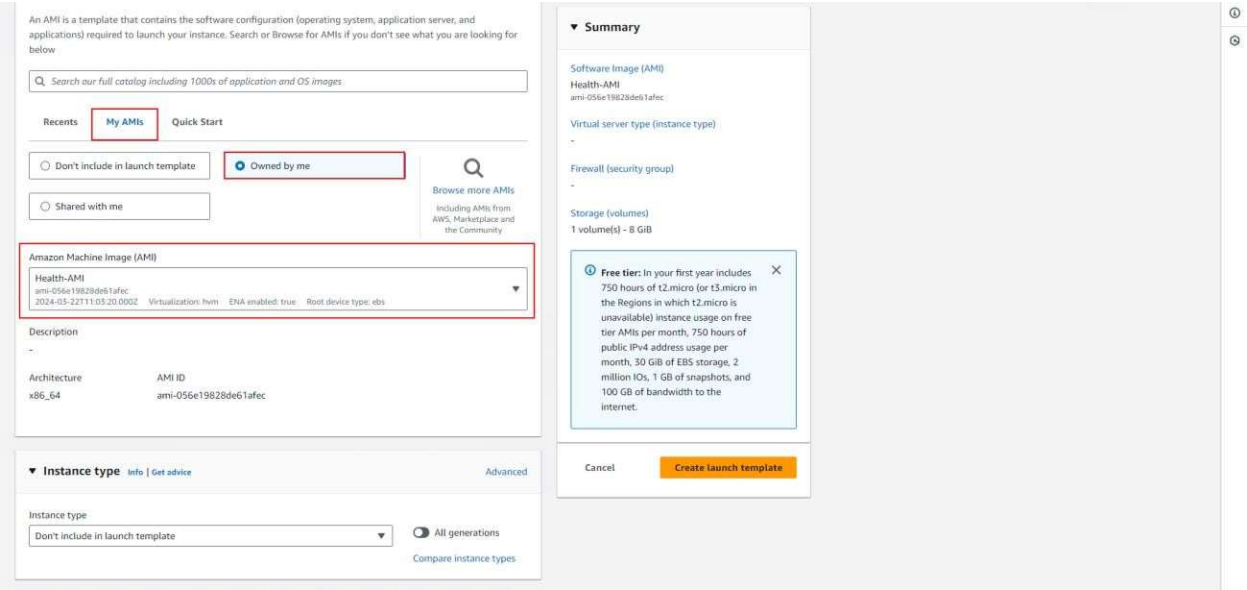
Then, click on the **Launch Templates** section and afterward click on the **Create Launch Template** button.



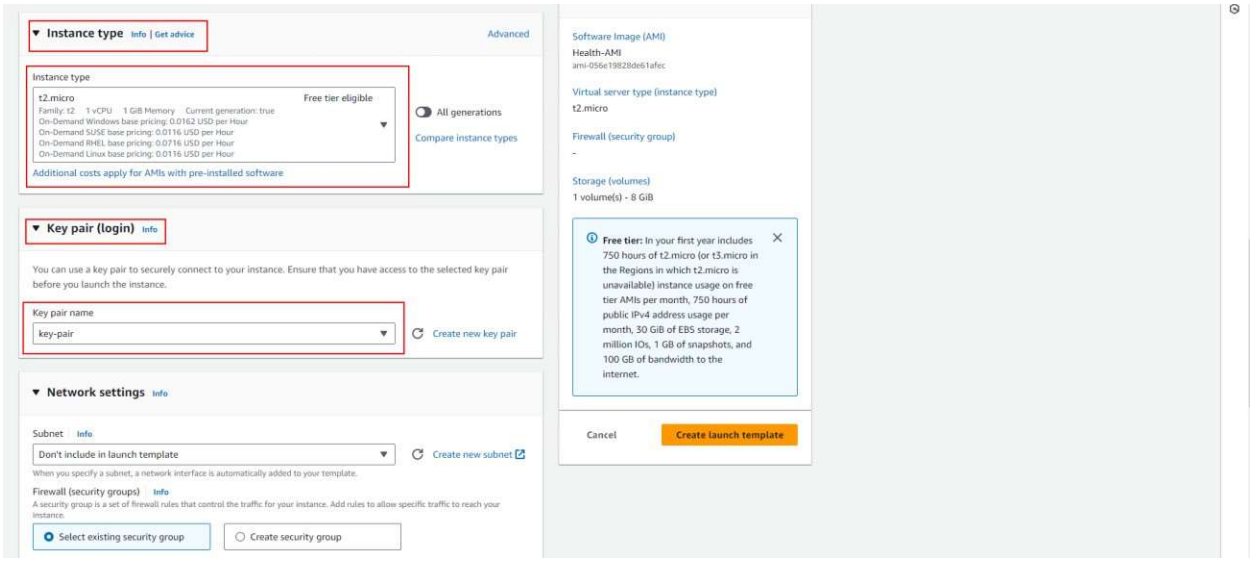
At this stage, specify a **name** and **version** for the template.



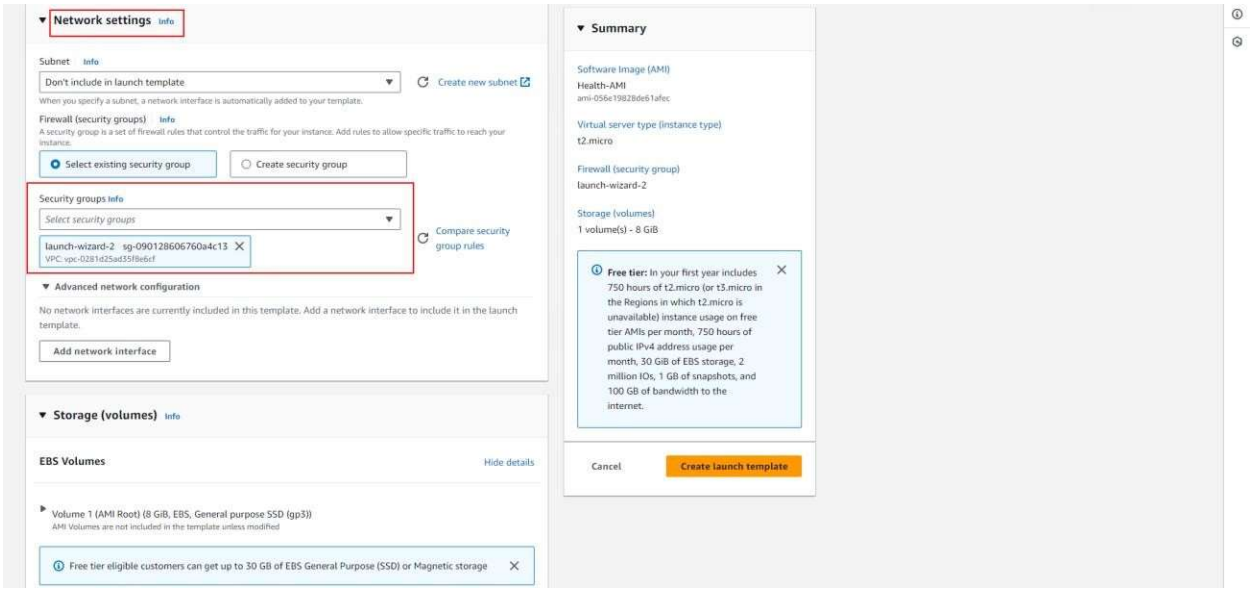
At this stage, you need to select your **AMI**.



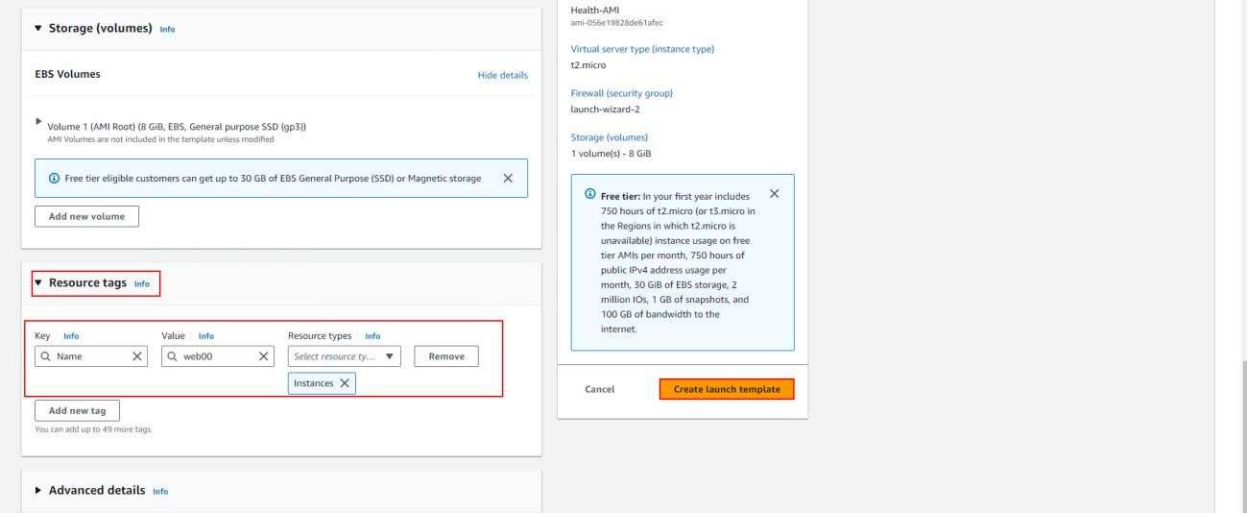
At this stage, you need to specify the **Instance Type** and the **Key Pair**.



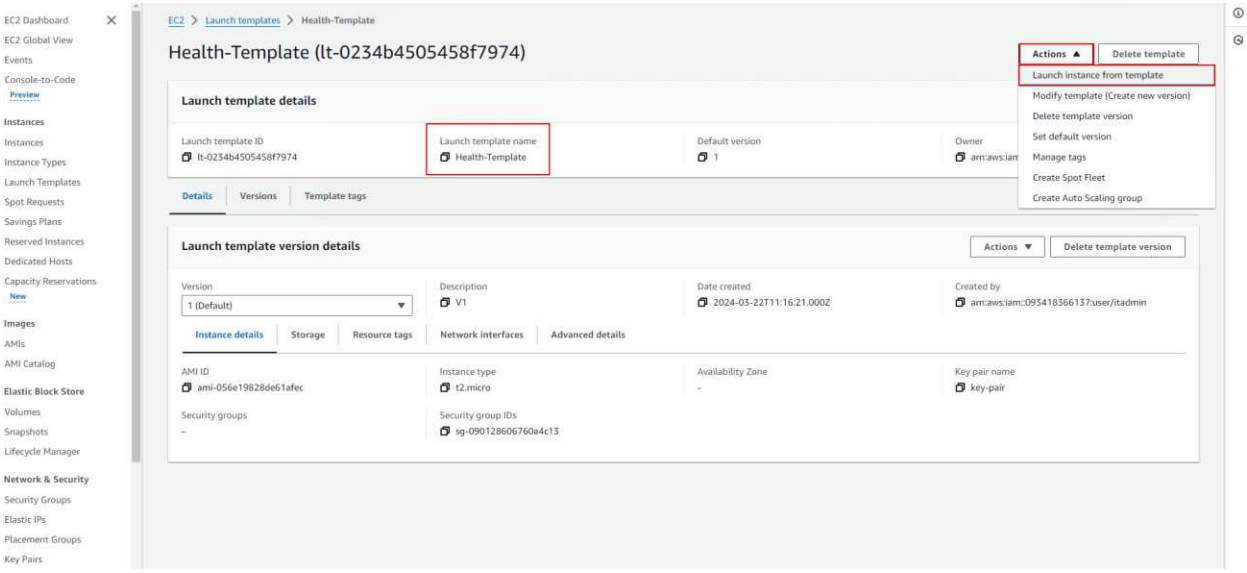
At this stage, you need to configure the **Security Group**.



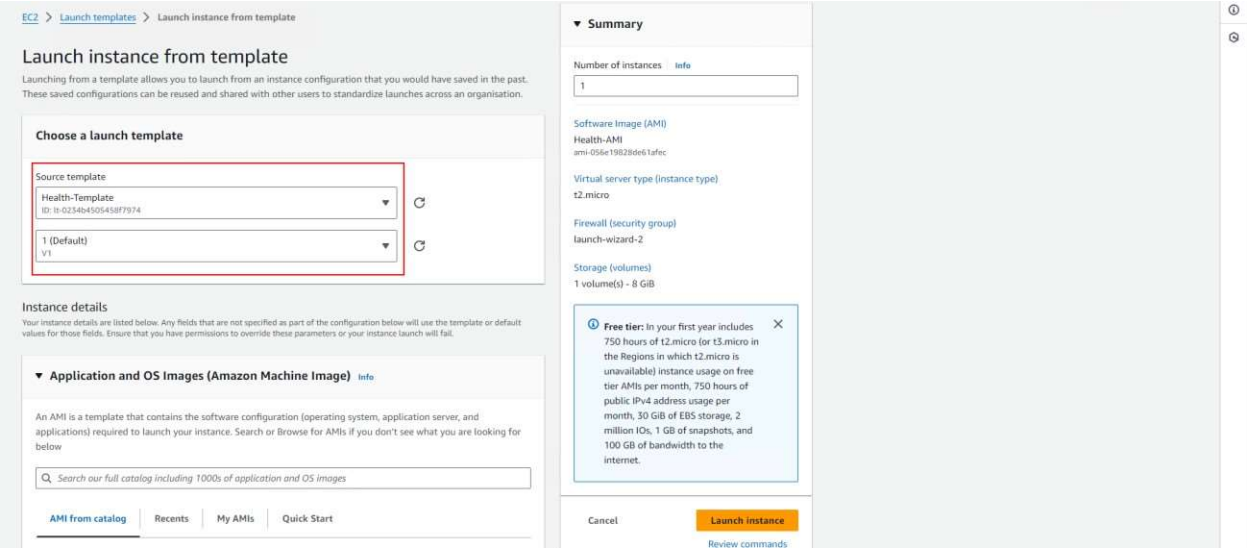
At this stage, define a **Tag** for the template.



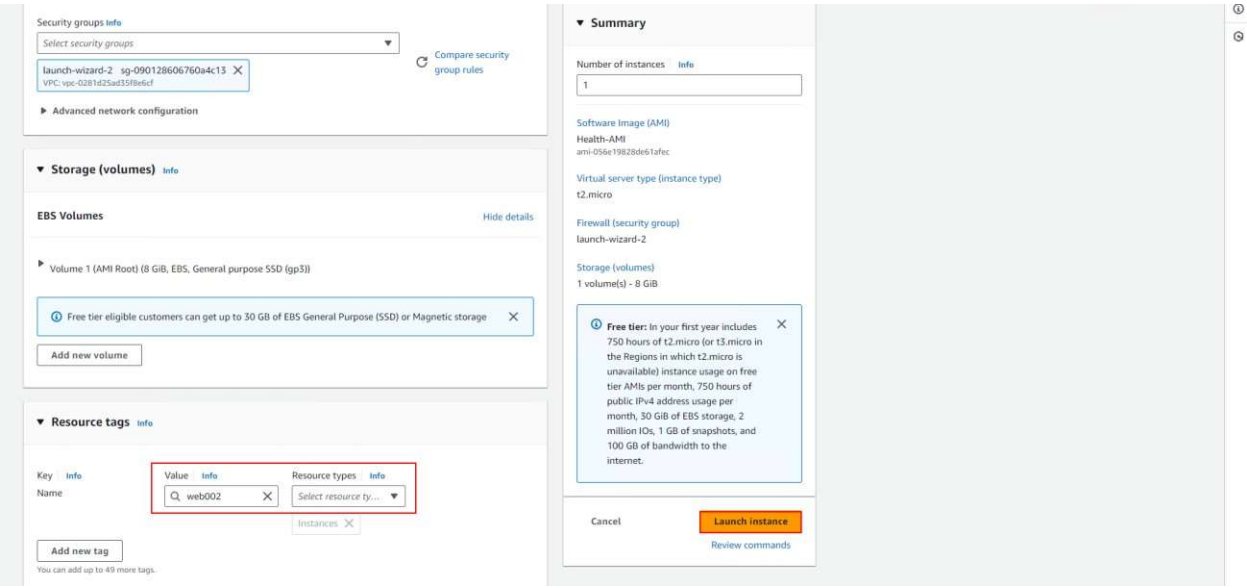
Then, at this stage, select your **template** and from the **Actions** menu, choose **Launch Instance from Template**.



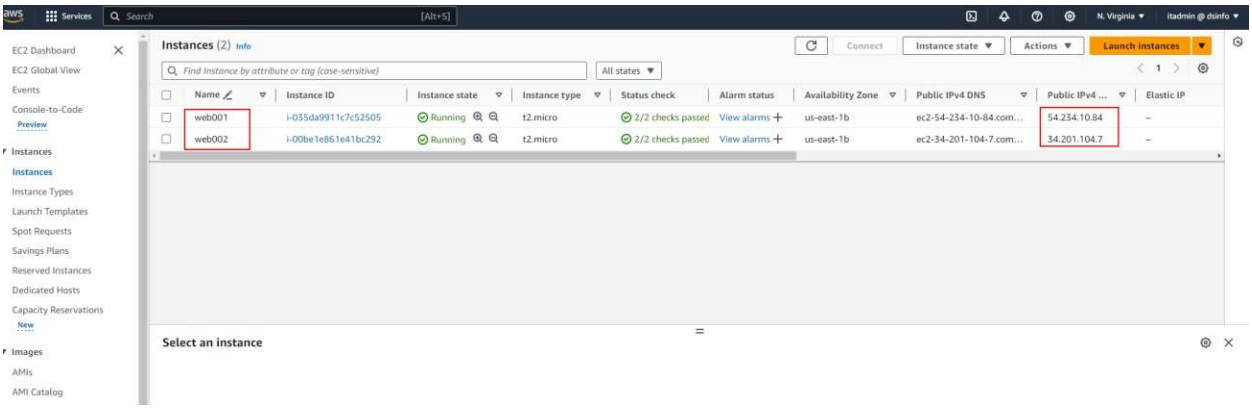
At this stage, specify the **Source Template**.



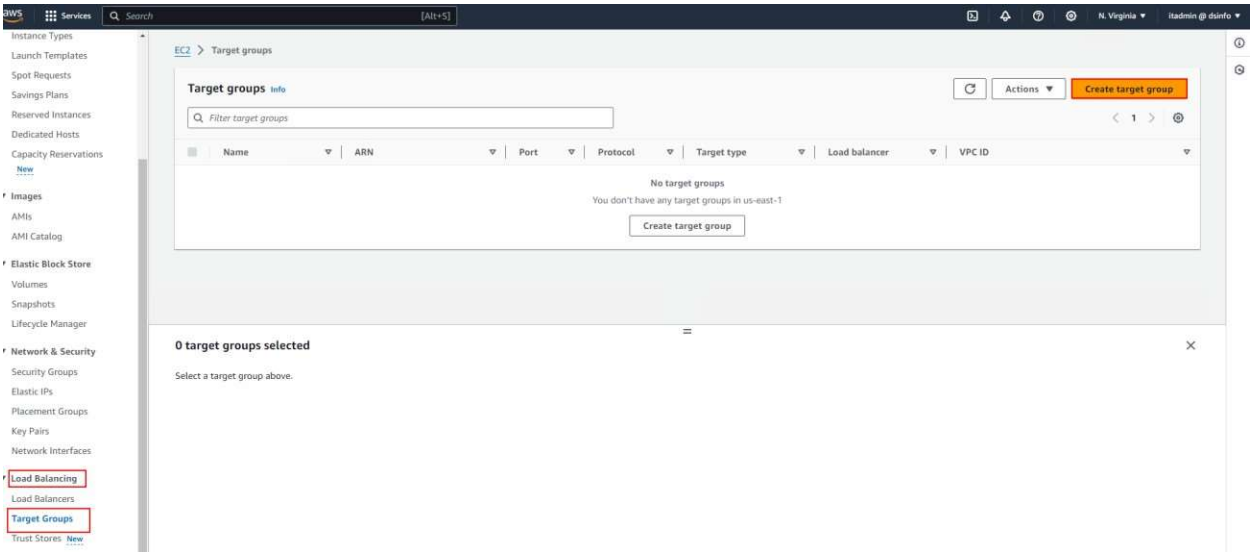
At this stage, assign a **Tag** to the instance and then click the **Launch Instance** button.



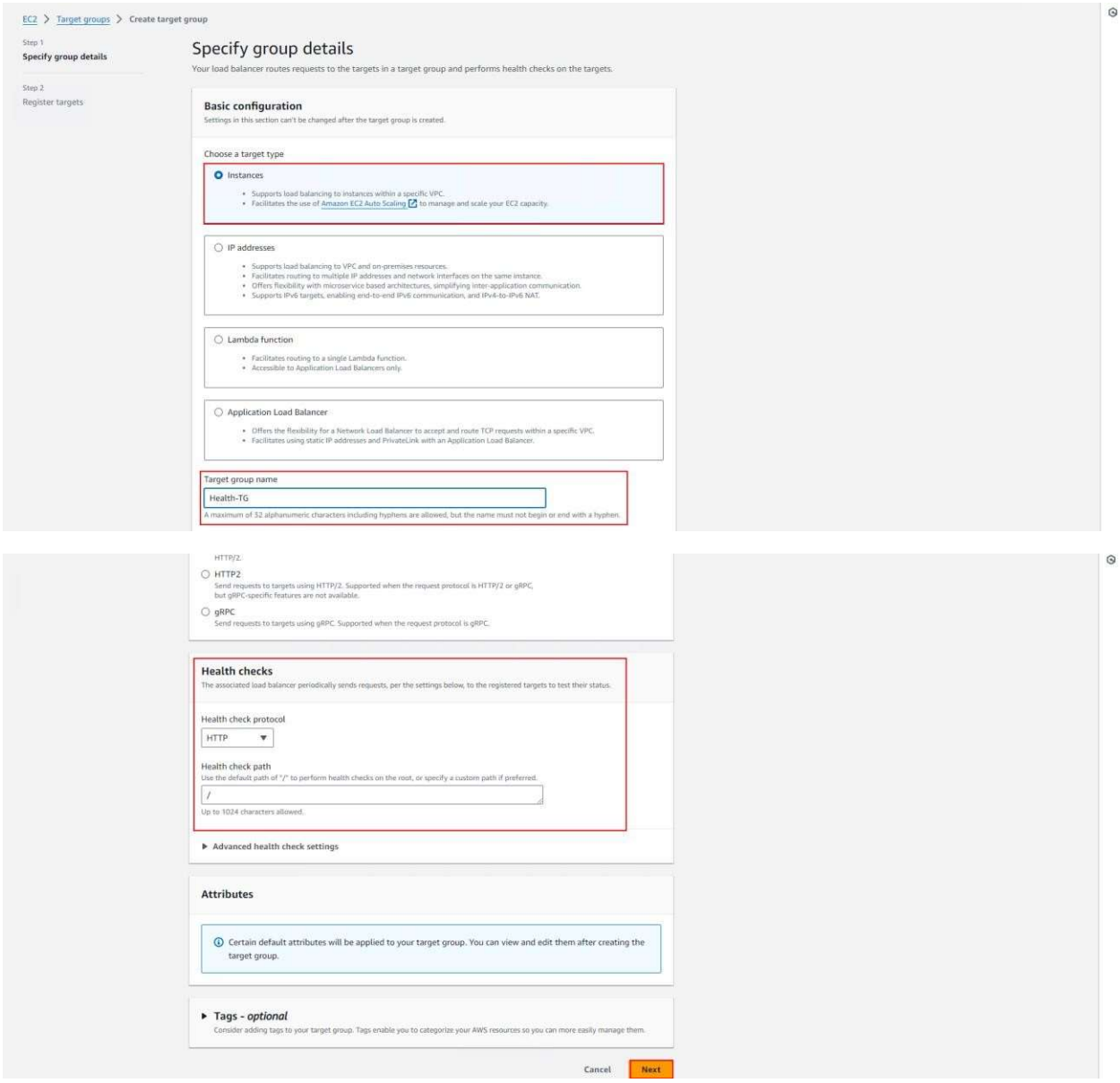
As shown in the image below, a **new instance** has been created from our **template**.



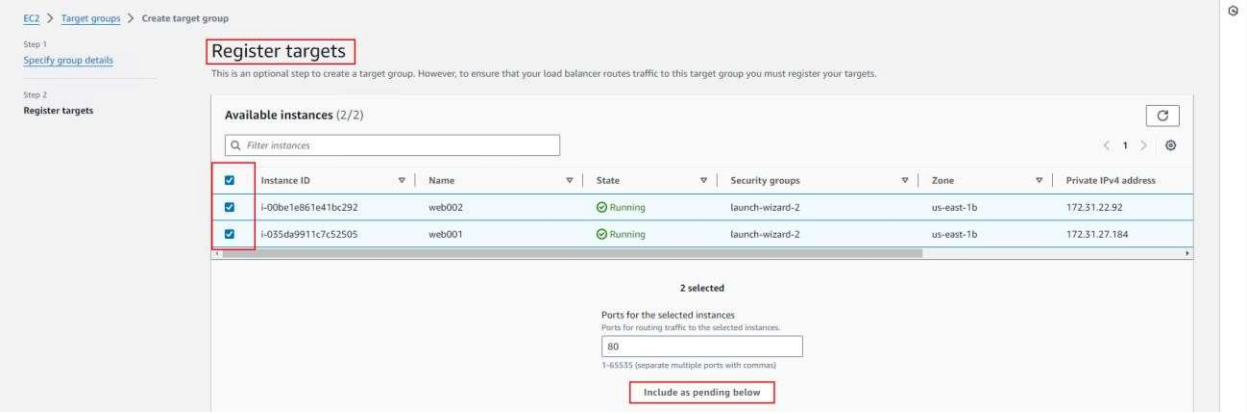
At this stage, click on the **Load Balancing** section, then click on **Target Groups**, and afterward click the **Create Target Group** button.



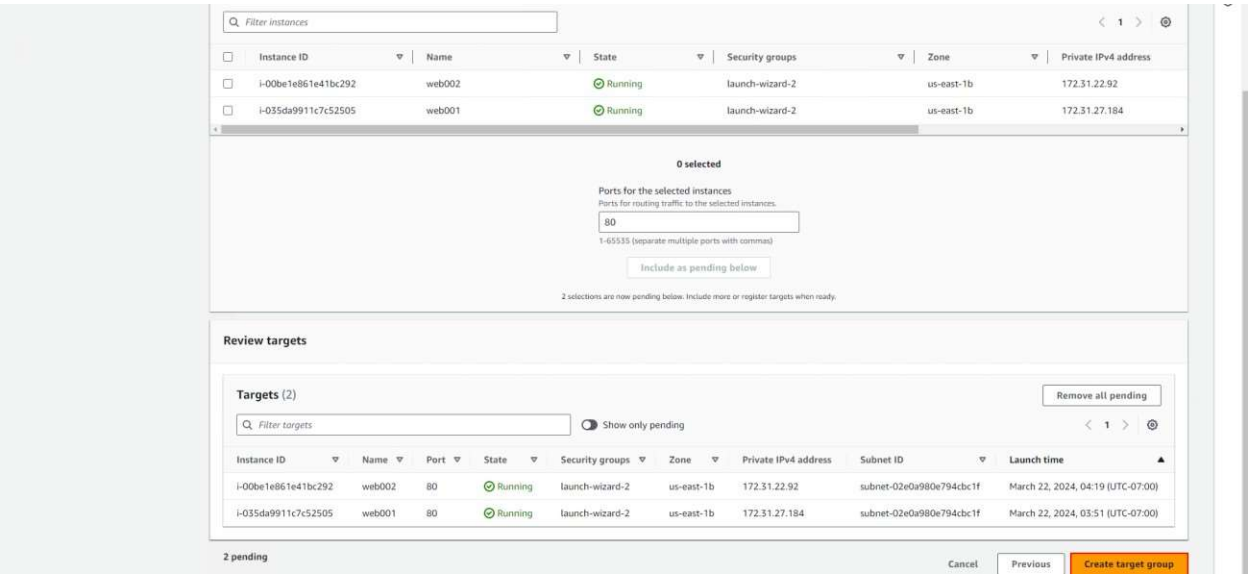
At this stage, select the following options as shown in the image, and then click the **Next** button.



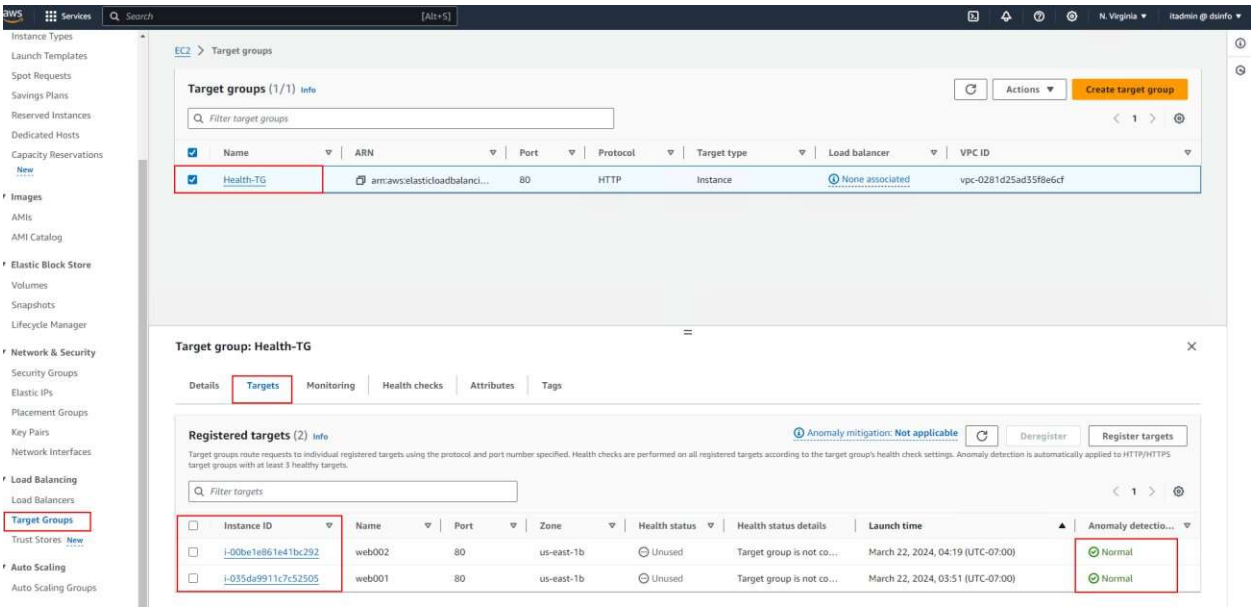
At this stage, click on the **Include as Pending Below** button.



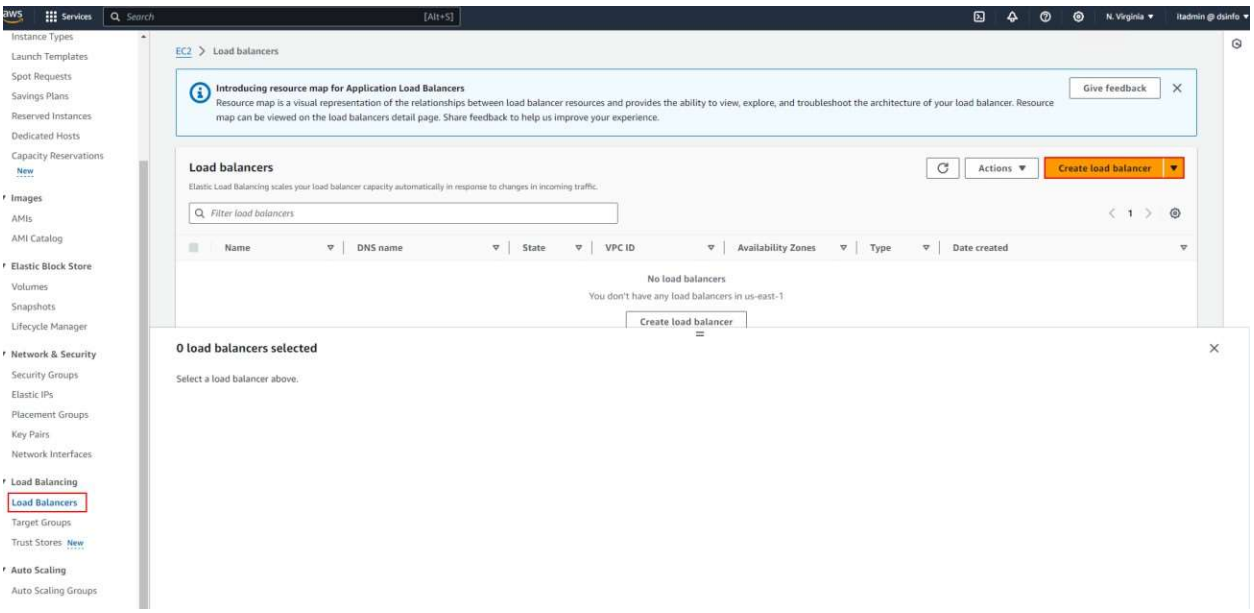
Then, at this stage, click on the **Create Target Group** button.



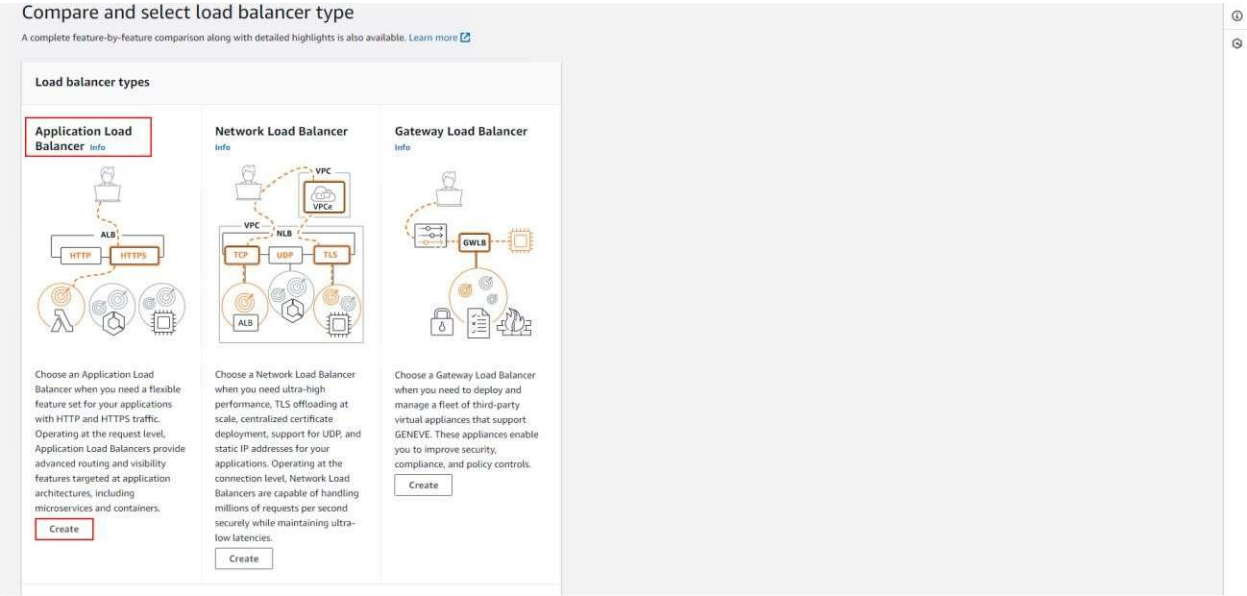
As shown in the image below, the instances in the **Target Group** are in a **healthy** state.



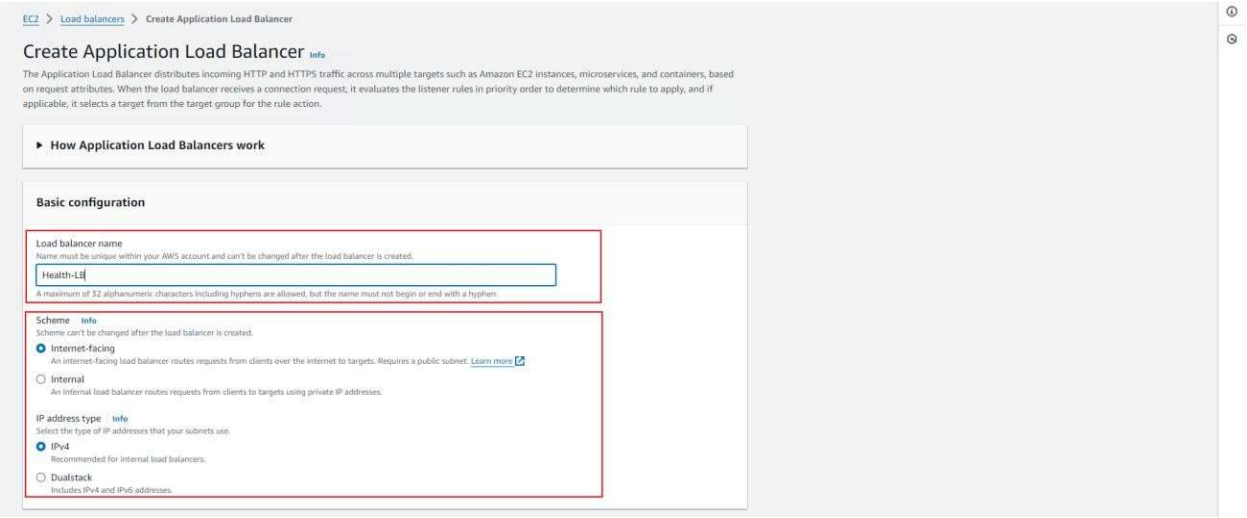
At this stage, click on the **Load Balancers** section, and then click the **Create Load Balancer** button.



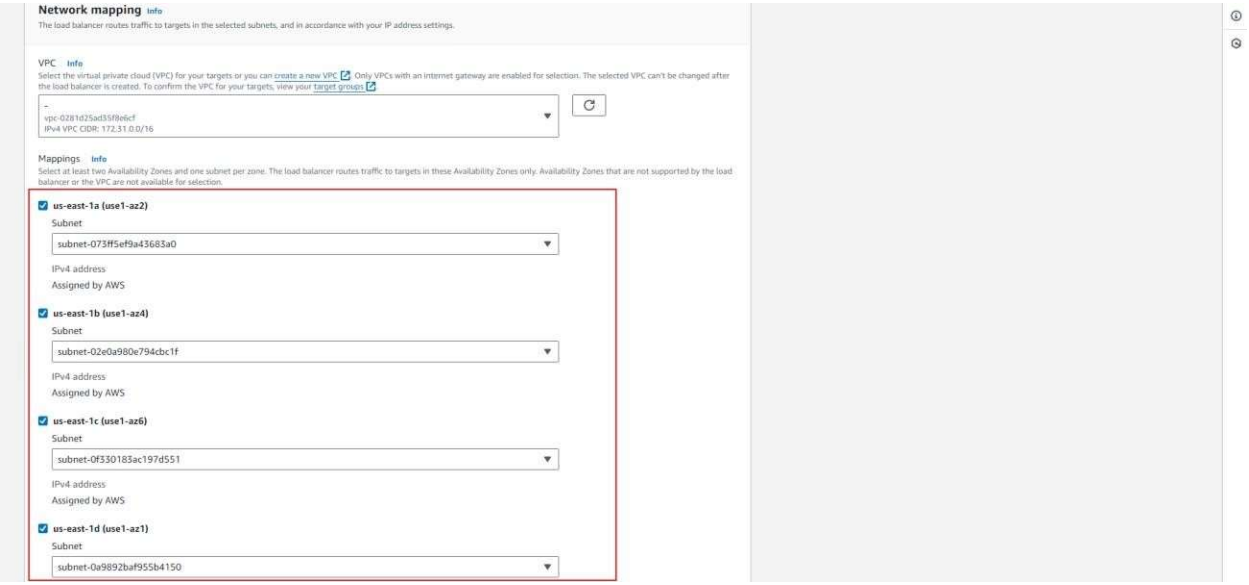
At this stage, select **Application Load Balancer**, then click on the **Create** button.



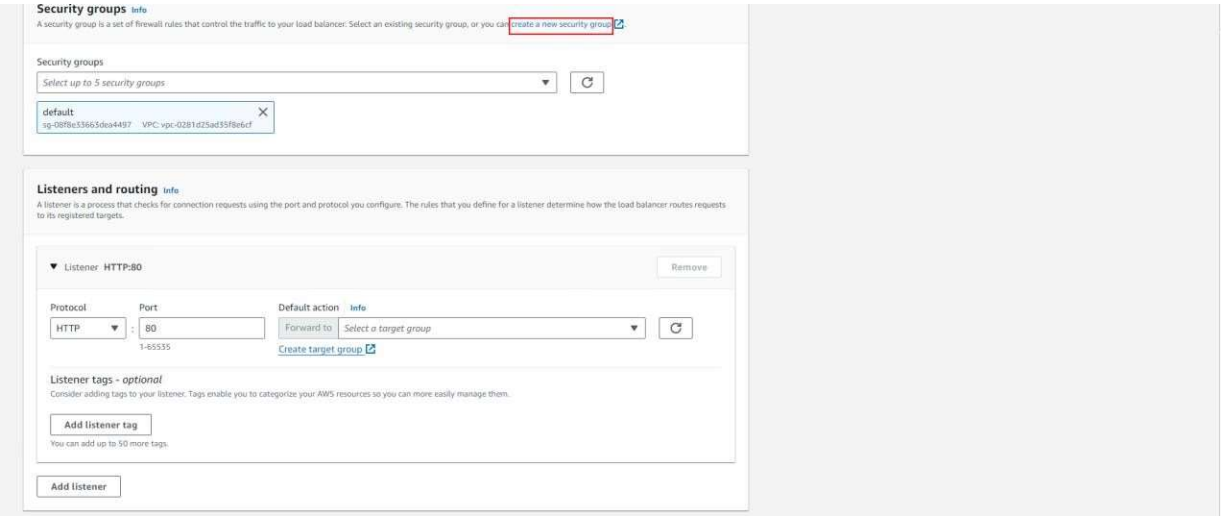
In this section, choose a **name** for the Load Balancer.



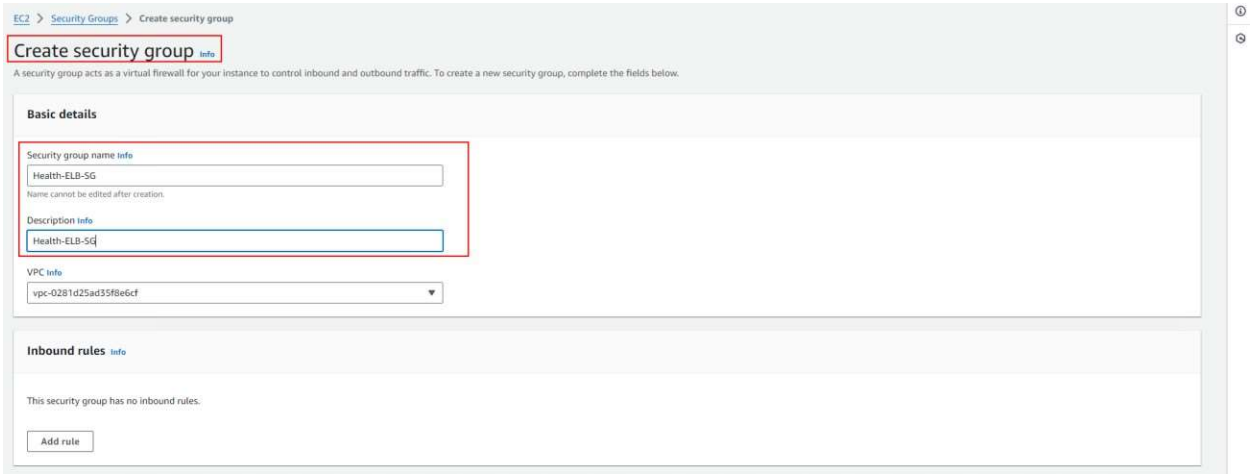
At this stage, select **all Availability Zones** to increase **availability**.



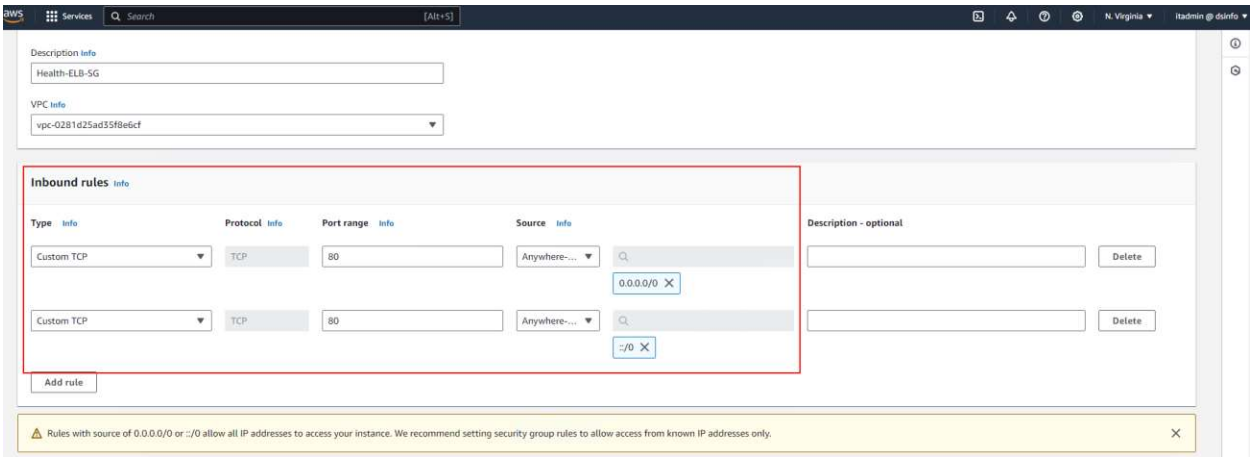
At this stage, you need to select a **Security Group** for the Load Balancer.

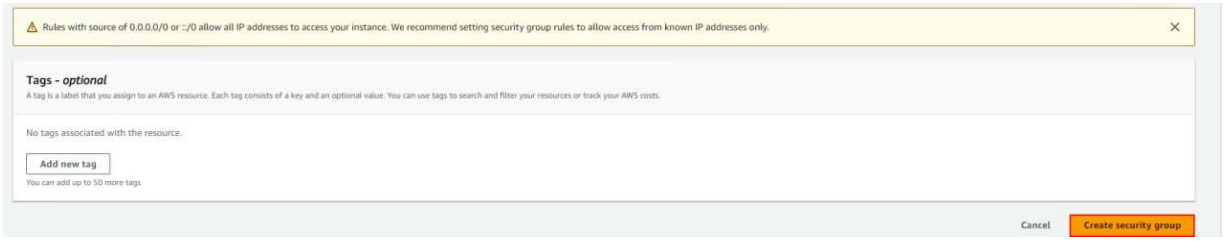


At this stage, specify a **name** for the Security Group.

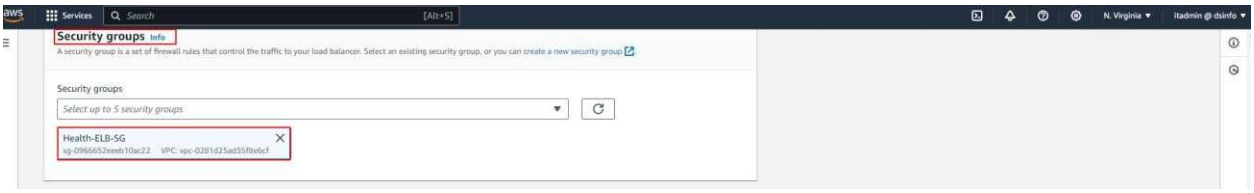


In the **Inbound Rule** section, you need to create the following rules, and then click on the **Create Security Group** button.

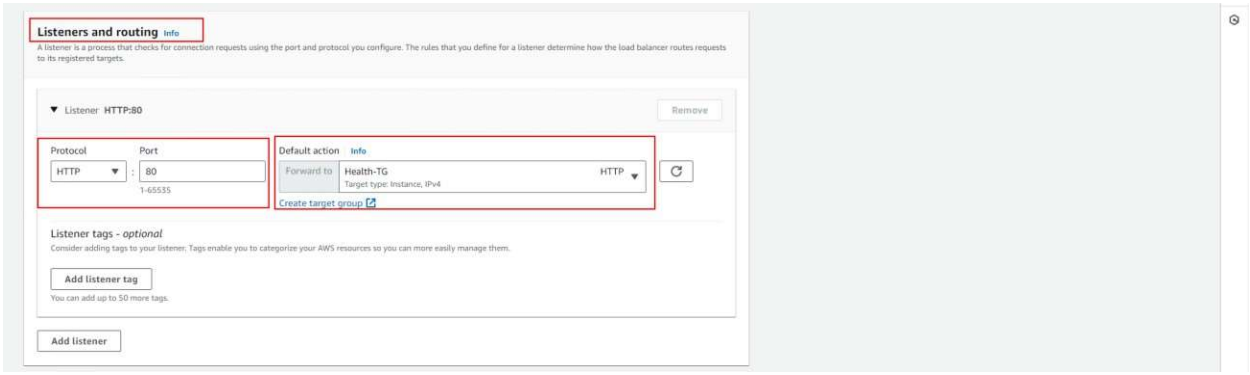




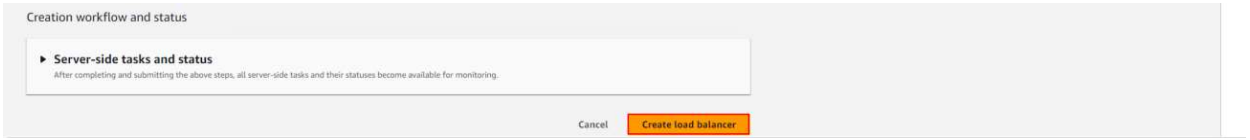
At this stage, select the **Security Group** you have created.



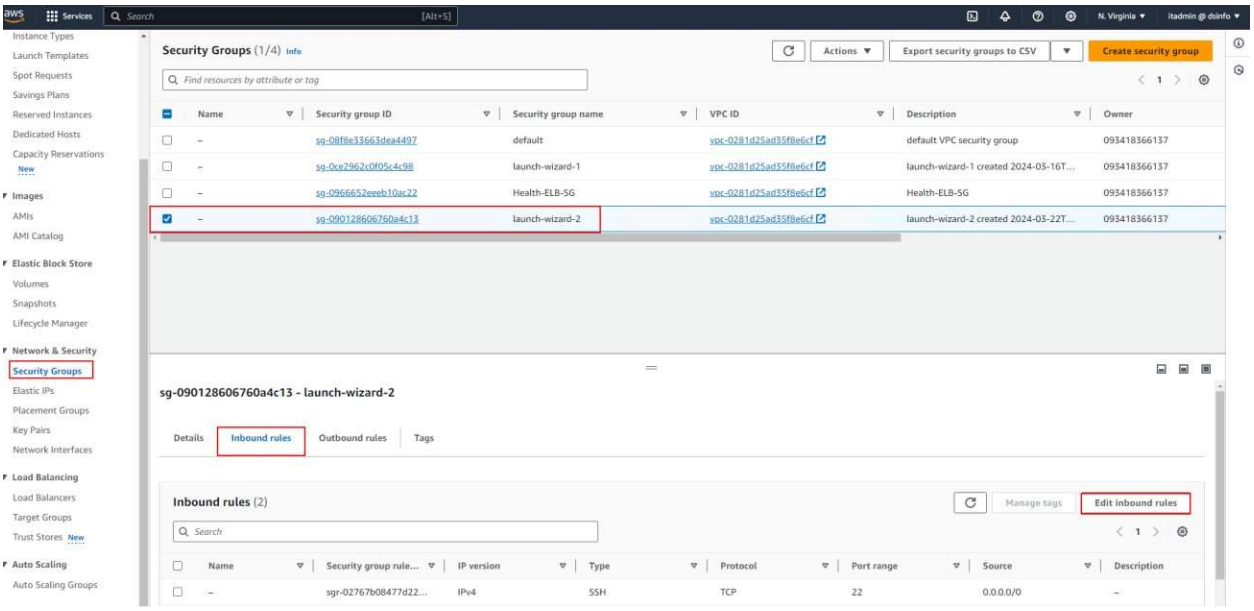
In the **Listener** section, set the **Default Action** to your **Target Group**.



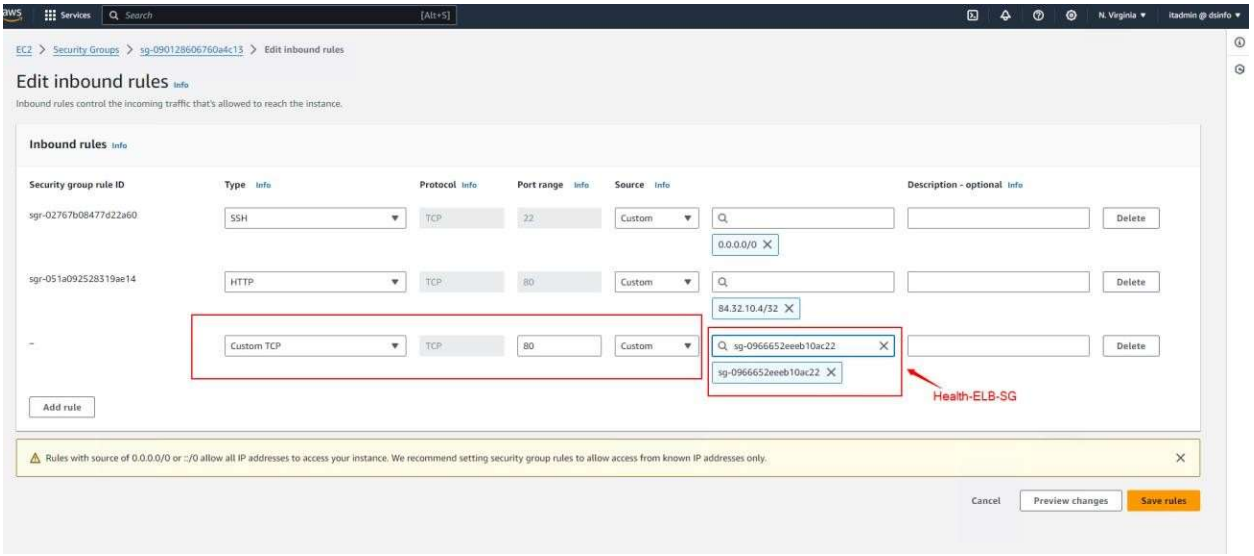
And finally, click on **Create Load Balancer**.



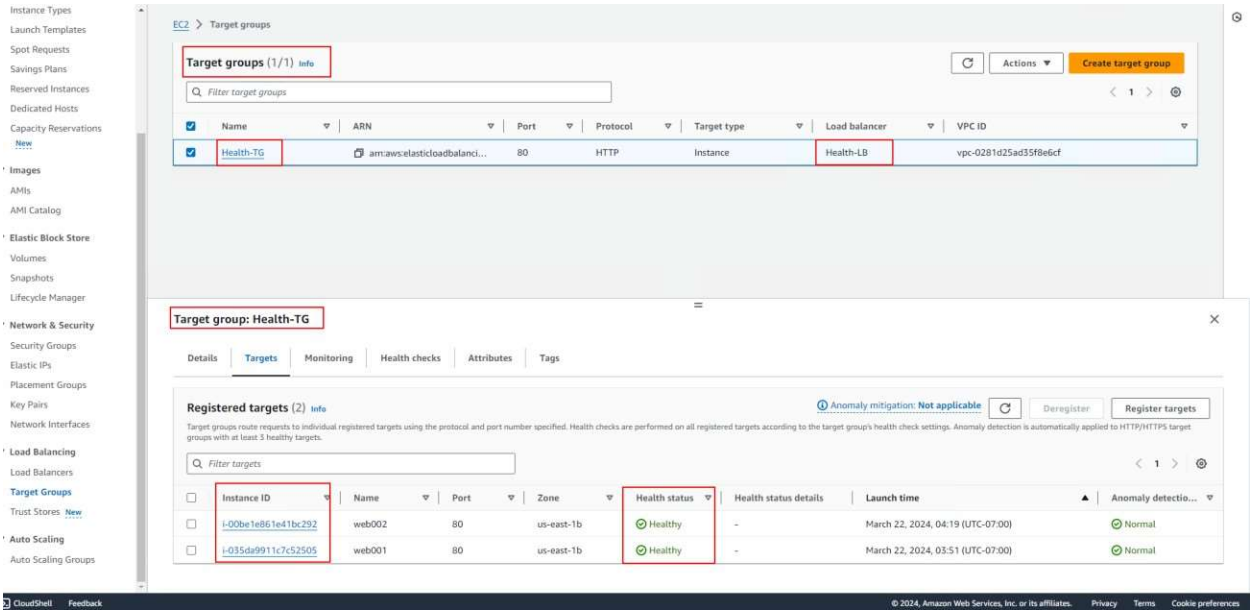
At this stage, you need to select the **Security Group** associated with your instances and click on the **Edit Inbound Rules** button in the **Inbound Rules** section.



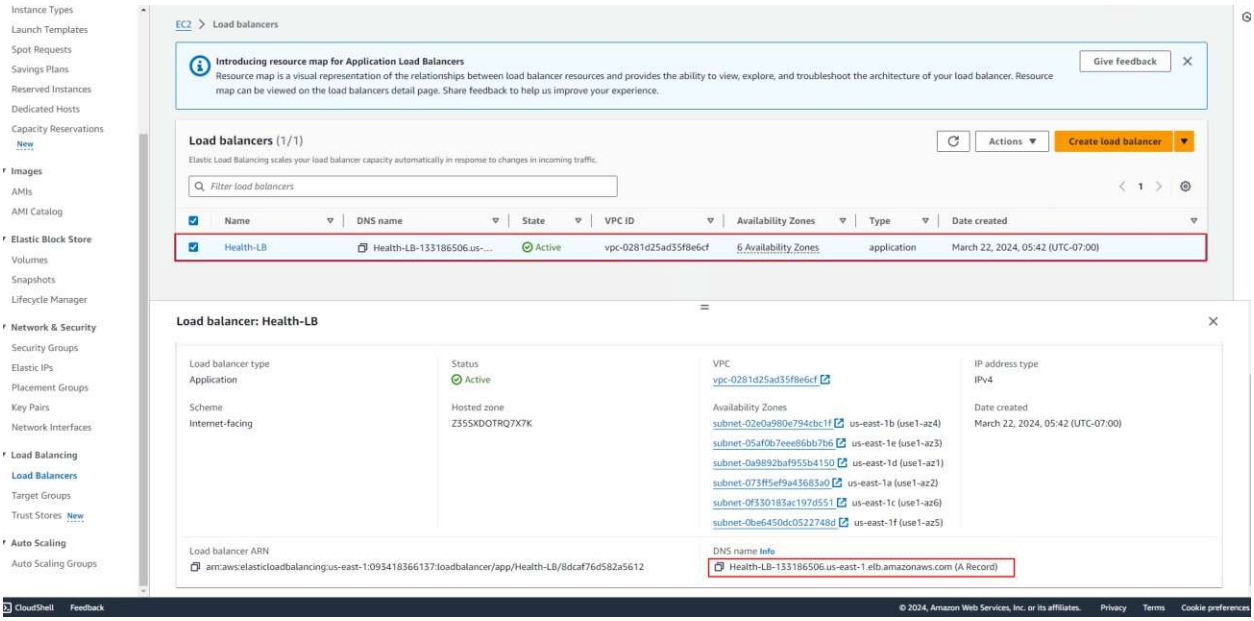
Create a new **rule** of type **TCP** on **port 80**, and set its **source** to the **Security Group of the ELB**.



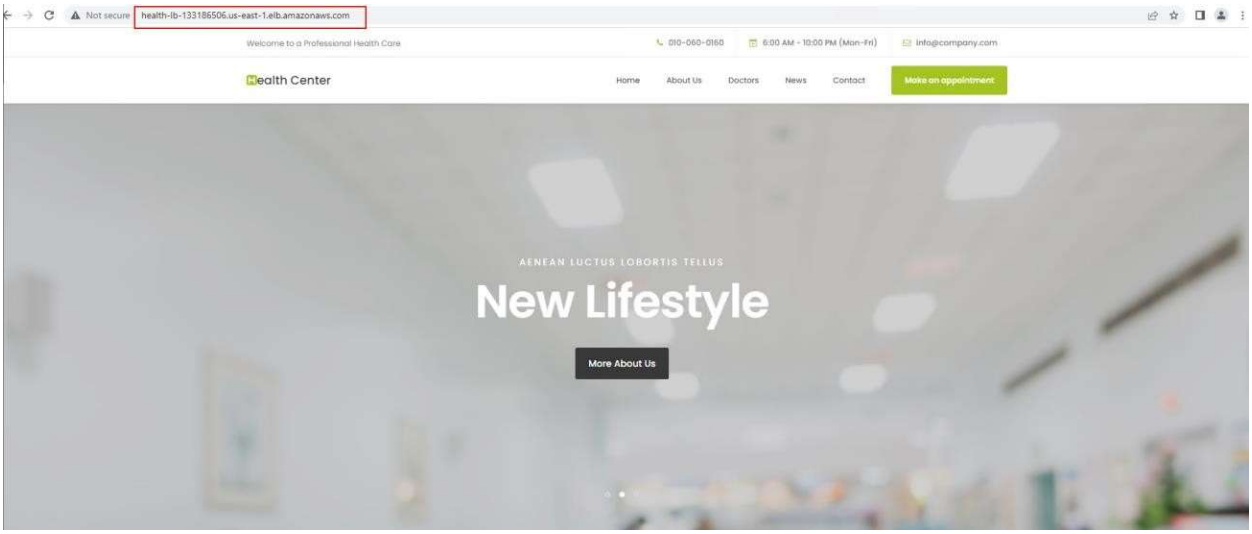
As shown in the image below, the status of the instances in the **Target Group** changes to **Healthy**.



At this stage, click on the **Load Balancer** you created and then copy its **DNS Name** link.



As shown in the image below, the **Load Balancer** is working correctly and is routing traffic to the **EC2 instances**.



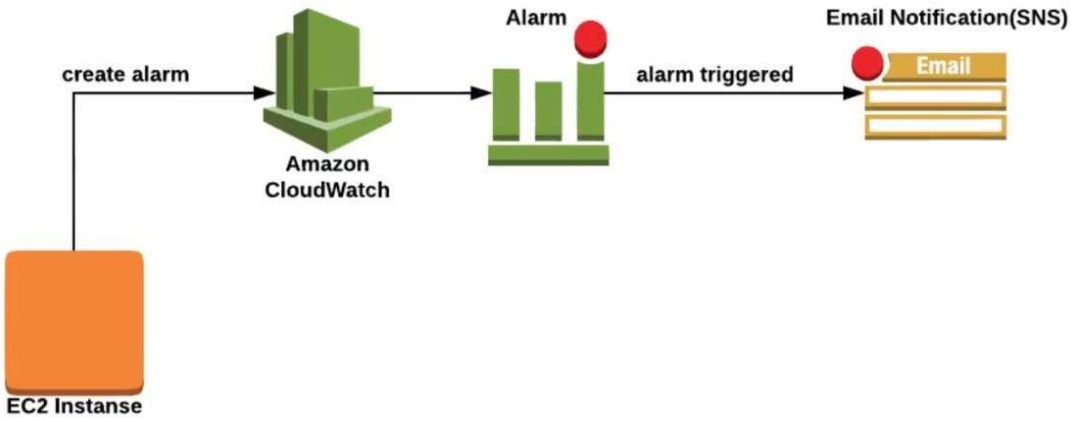
Introduction to AWS CloudWatch

AWS CloudWatch is a **monitoring service**; it is also a **logging solution**.

Use Cases of AWS CloudWatch:

- **Monitoring AWS Resources** such as EC2, RDS, Lambda, EBS, etc.
- **Collecting and tracking metrics** like CPU utilization, memory usage, and disk I/O
- **Setting alarms** to automatically trigger actions when certain thresholds are exceeded
- **Aggregating and storing logs** from applications and AWS services
- **Creating dashboards** for real-time performance visualization
- **Troubleshooting operational issues** by analyzing logs and metrics
- **Automating responses** using CloudWatch Events or Alarms (e.g., stop an instance or send an SNS notification)

In the image below, you can see how **CloudWatch** works.



By default, **CloudWatch** collects data for each metric every **5 minutes** and displays it on graphs. You can reduce this interval to **1 minute**, but there is an additional cost.
To enable 1-minute monitoring, go to the **Manage Detailed Monitoring** section and enable the feature.

Introduction to AWS EFS

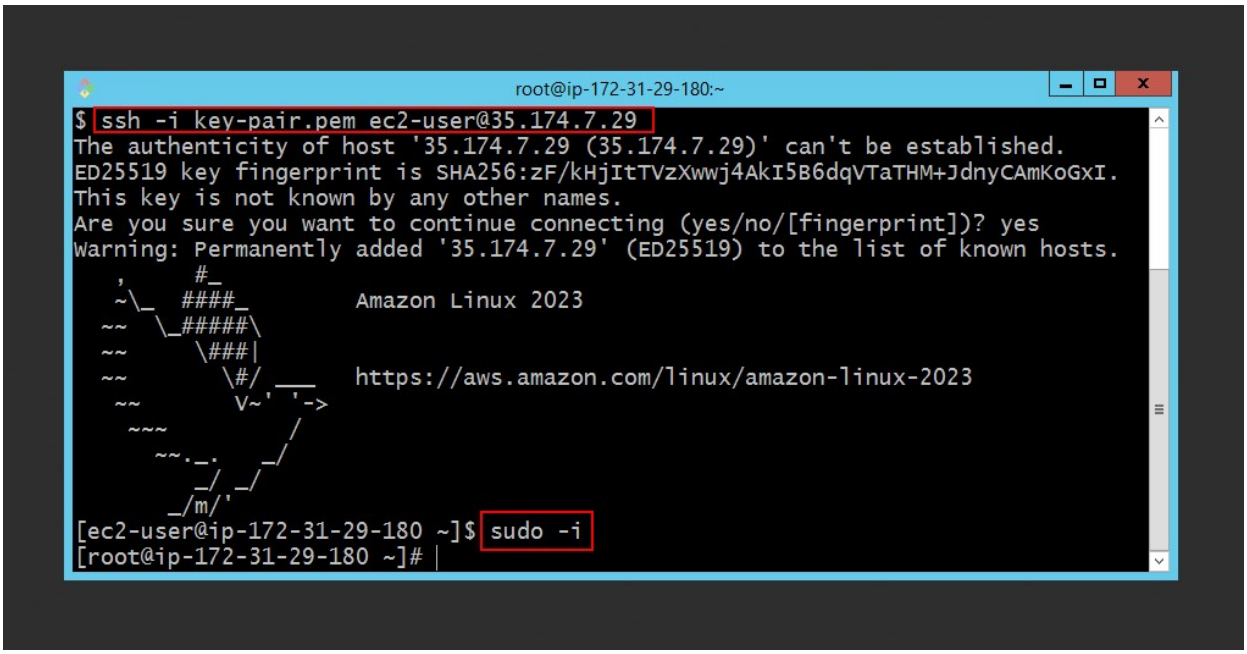
Introduction to AWS EFS (Elastic File System)

AWS **EFS** is a fully managed, scalable, and elastic **network file system** designed for use with AWS Cloud services and on-premises resources. Key features include:

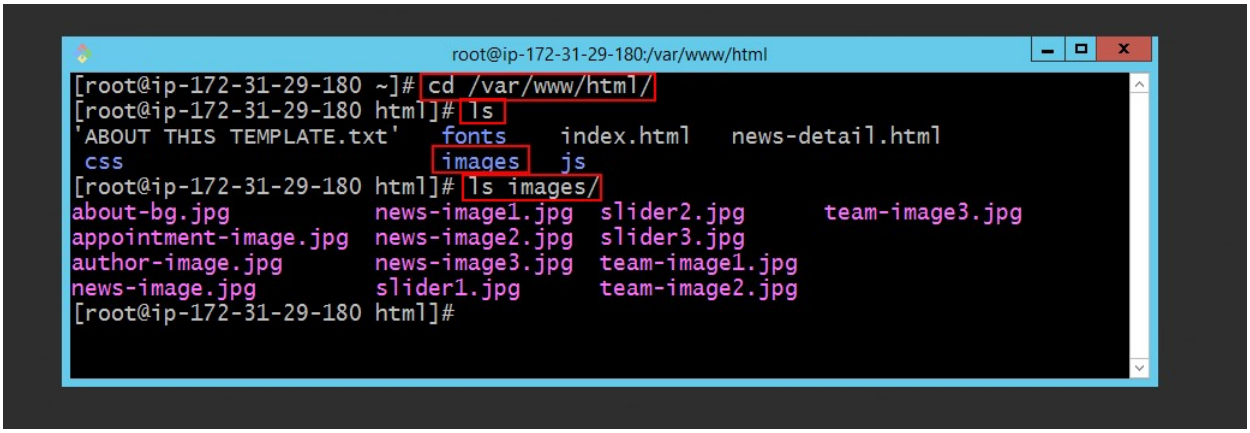
- **Scalable storage** that grows and shrinks automatically as you add or remove files.
- **Shared access** across multiple **EC2 instances** (can be mounted simultaneously).
- Works over **NFS (Network File System)** protocol.
- Provides **high availability and durability** across multiple Availability Zones.
- Ideal for use cases such as **content management**, **web serving**, **home directories**, **development environments**, and **big data analytics**.

EFS is easy to use, supports standard file system semantics, and is accessible from thousands of EC2 instances concurrently.

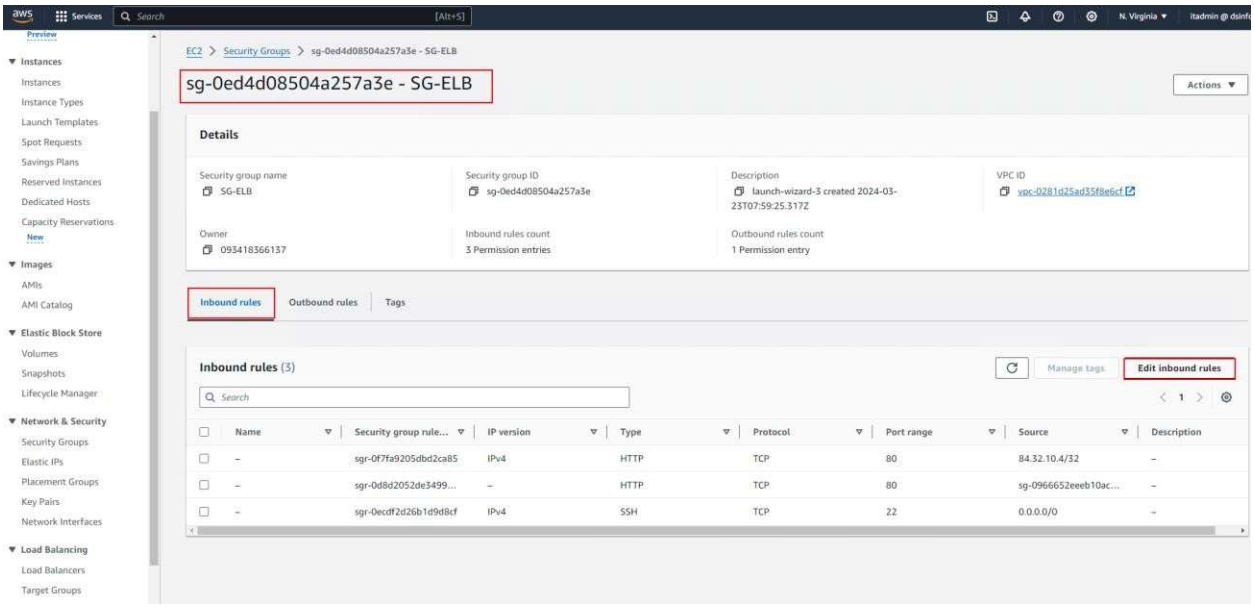
At this stage, first, log in to your **EC2 instance**.



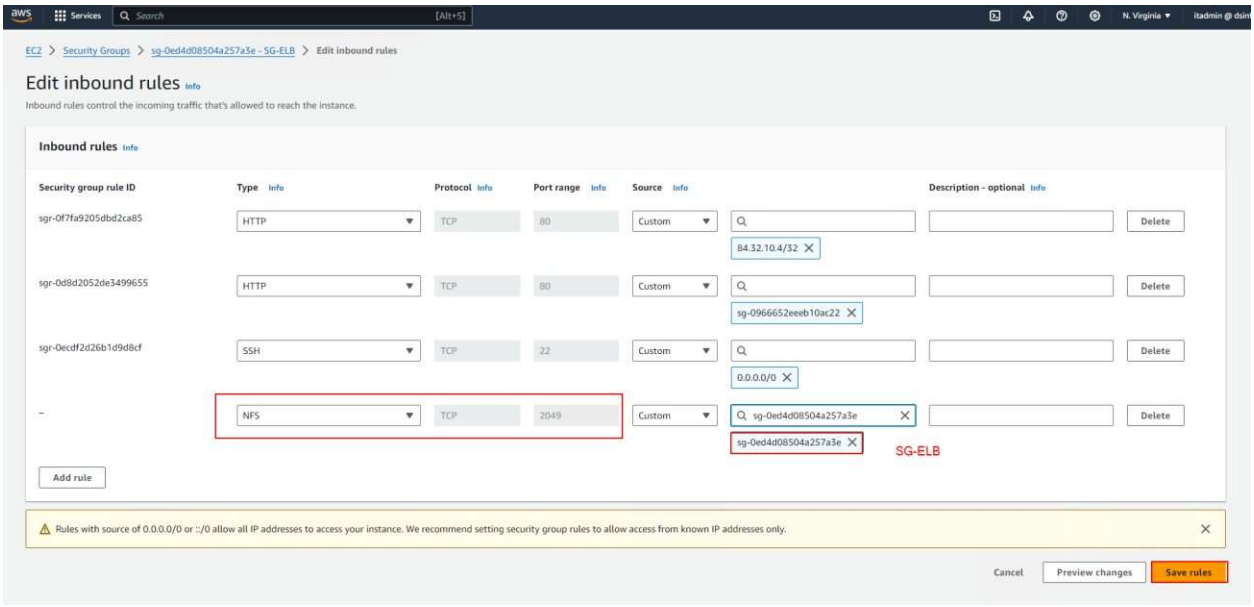
As seen in the **web server path**, our web service contains a folder named **Images**, which includes the website's images.



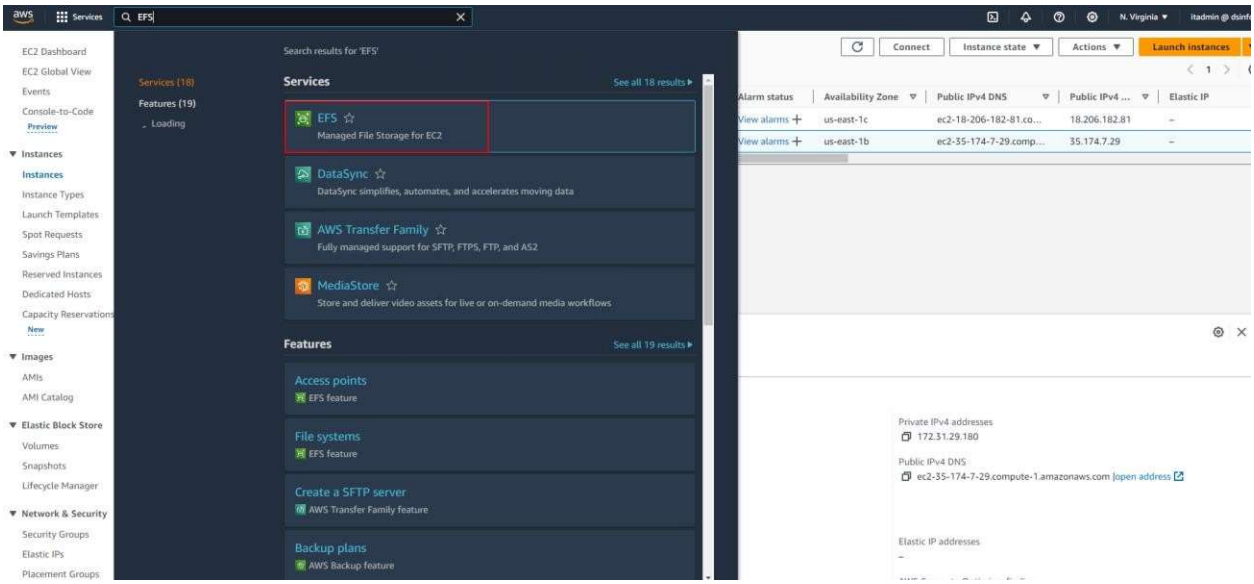
Go to the **ELB Security Group** section and then click on the **Edit Inbound Rules** button.



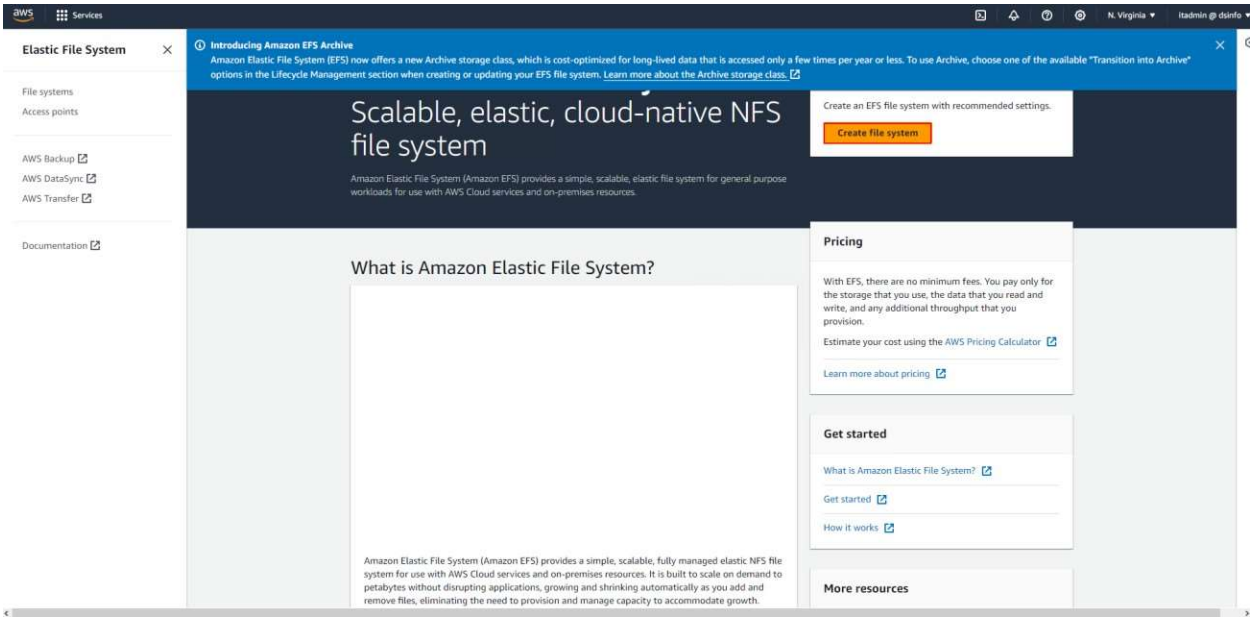
Create a new **rule** for the **NFS protocol**, set the **source** to the **ELB Security Group**, and then click on the **Save Rules** button.



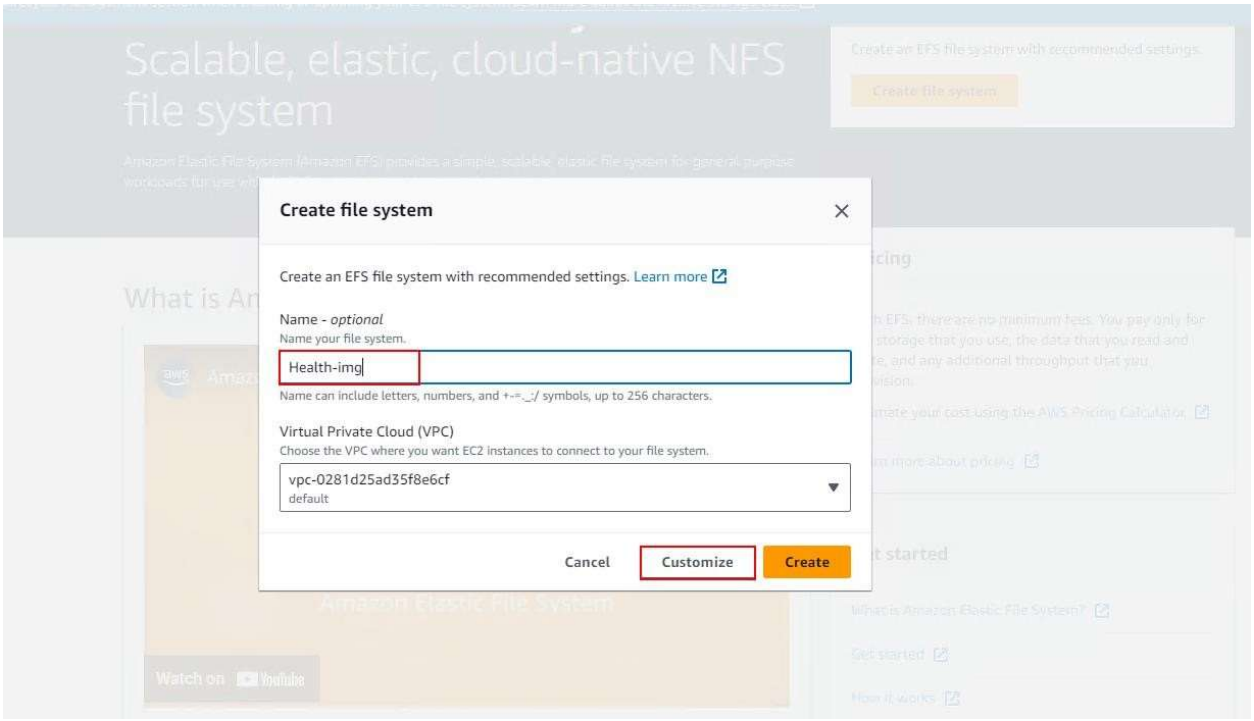
In the search bar, type **EFS**, then click on **EFS** from the results.



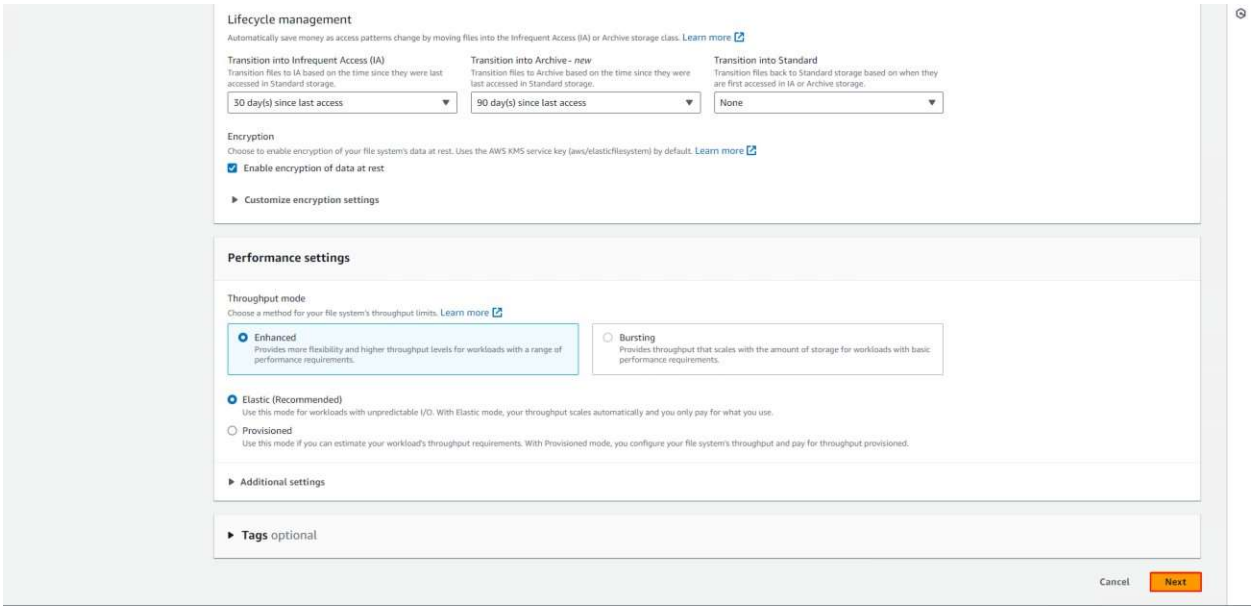
At this stage, click on the **Create File System** button.



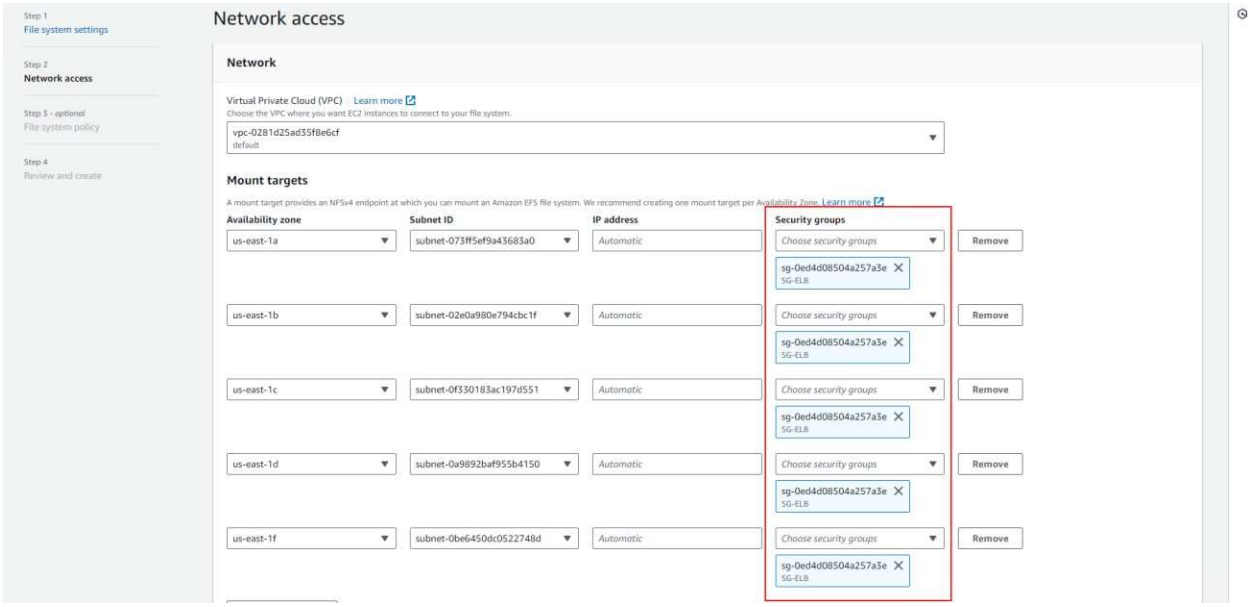
At this stage, choose a **name** for the **EFS** and select the **VPC**, then click on the **Customize** button.



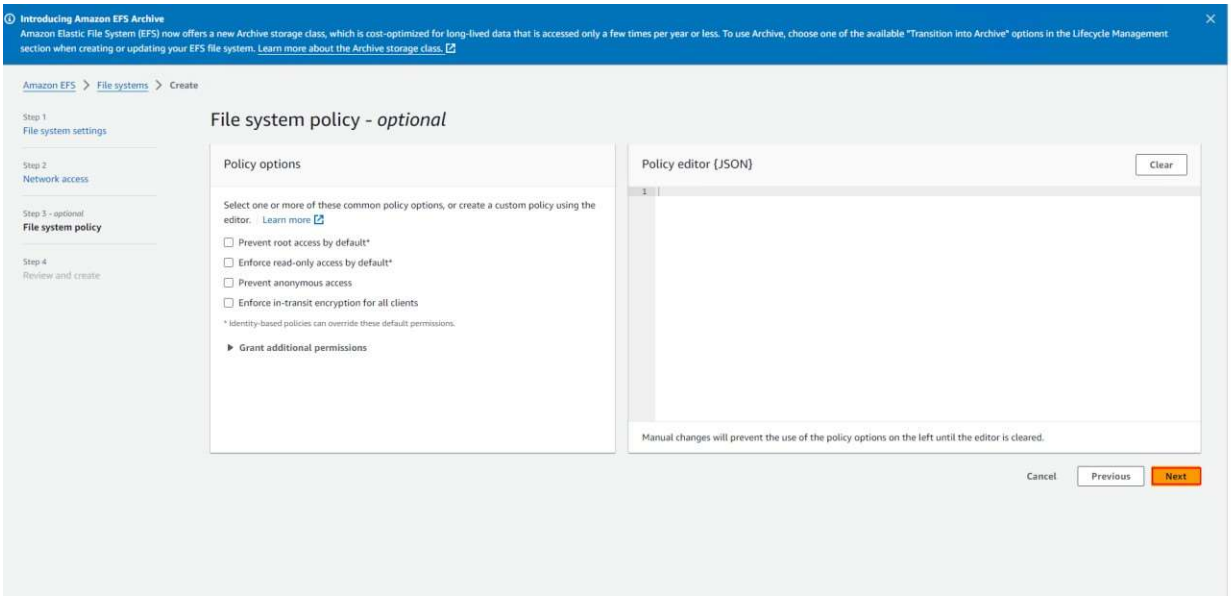
At this stage, click on the **Next** button.



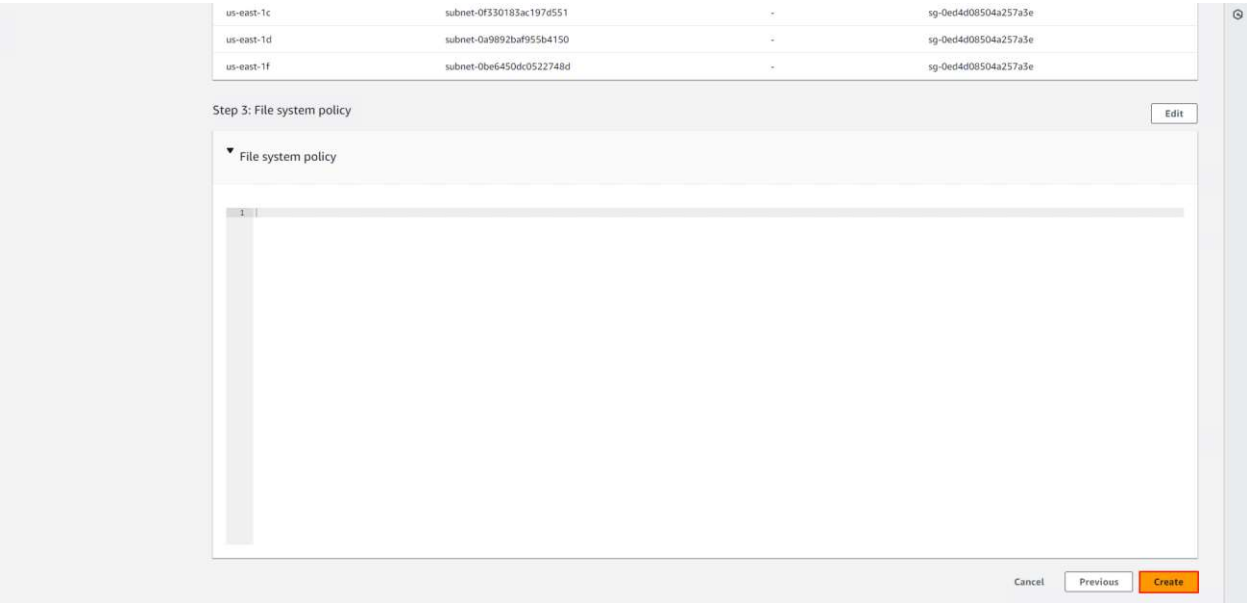
At this stage, set the **Security Group** for all **Availability Zones** to the **ELB Security Group**.



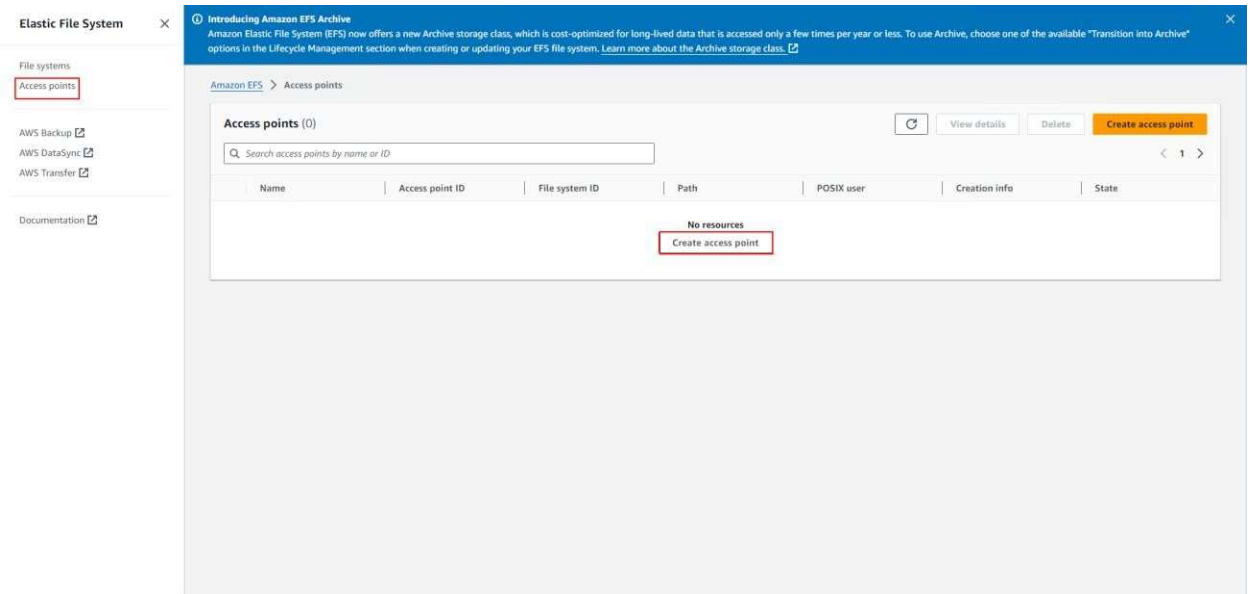
At this stage, click on the **Next** button.



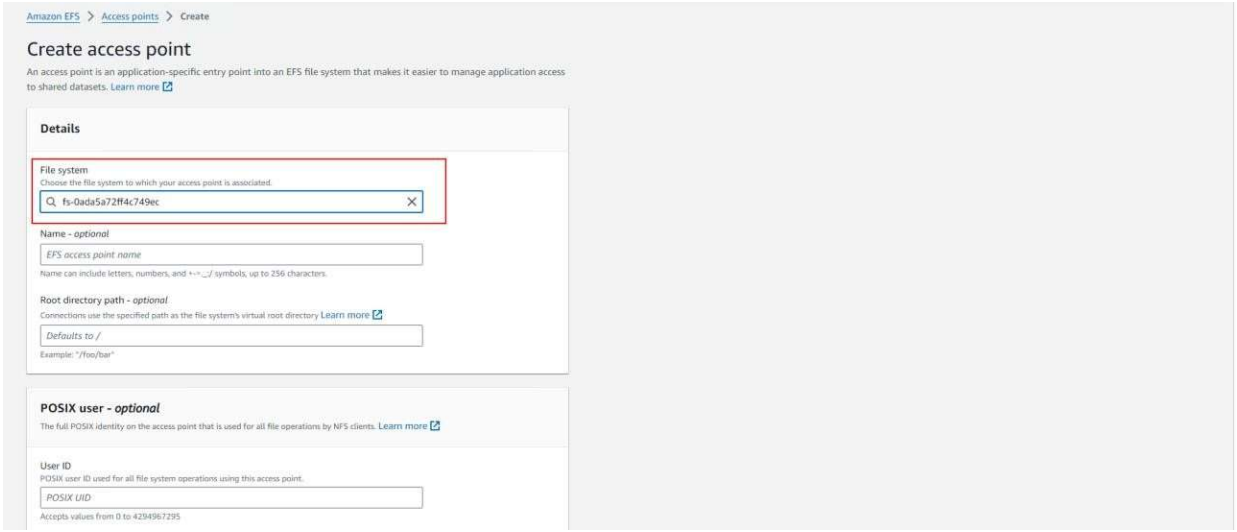
At this stage, click on the **Create** button.



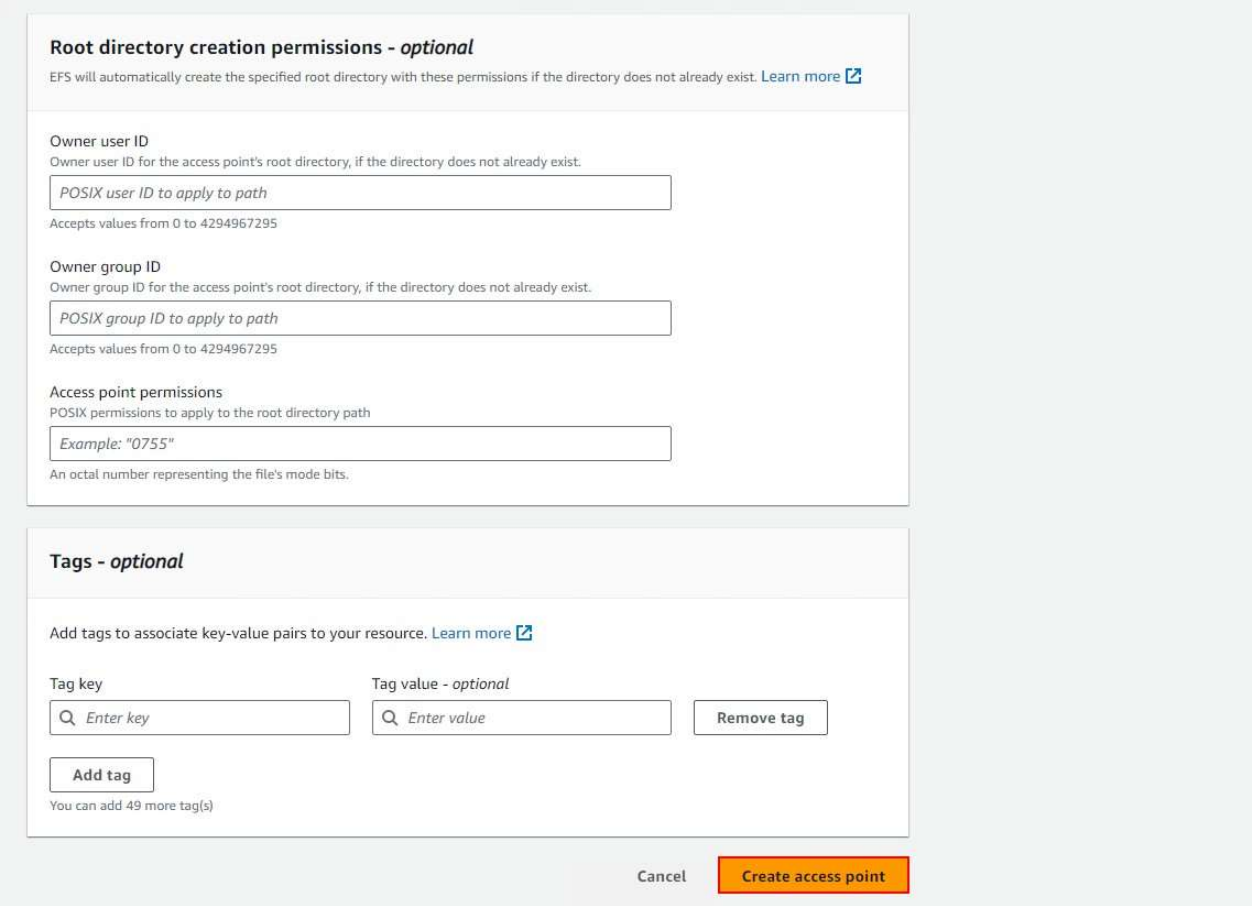
At this stage, click on the **Access Points** section, then click the **Create Access Point** button.



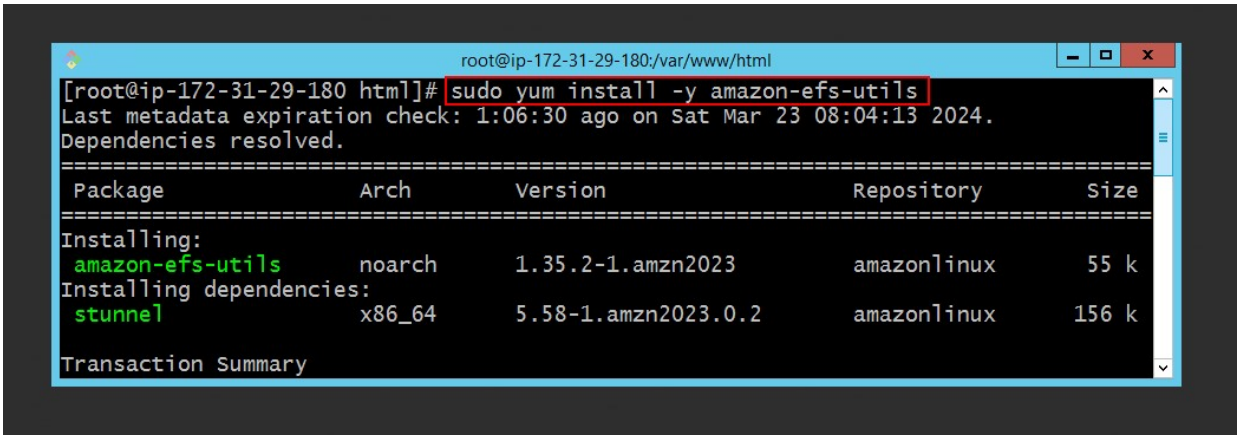
At this stage, you need to select the **EFS file system** that you created in the previous step.


Amazon EFS > Access points > Create
Create access point
An access point is an application-specific entry point into an EFS file system that makes it easier to manage application access to shared datasets. Learn more
Details
File system
Choose the file system to which your access point is associated.
fs-0ada5a72ff4c749ec
Name - optional
EFS access point name
Name can include letters, numbers, and +-=._:/ symbols, up to 256 characters.
Root directory path - optional
Connections use the specified path as the file system's virtual root directory Learn more
Defaults to /
Example: "/foo/bar"
POSIX user - optional
The full POSIX identity on the access point that is used for all file operations by NFS clients. Learn more
User ID
POSIX user ID used for all file system operations using this access point.
POSIX UID
Accepts values from 0 to 4294967295

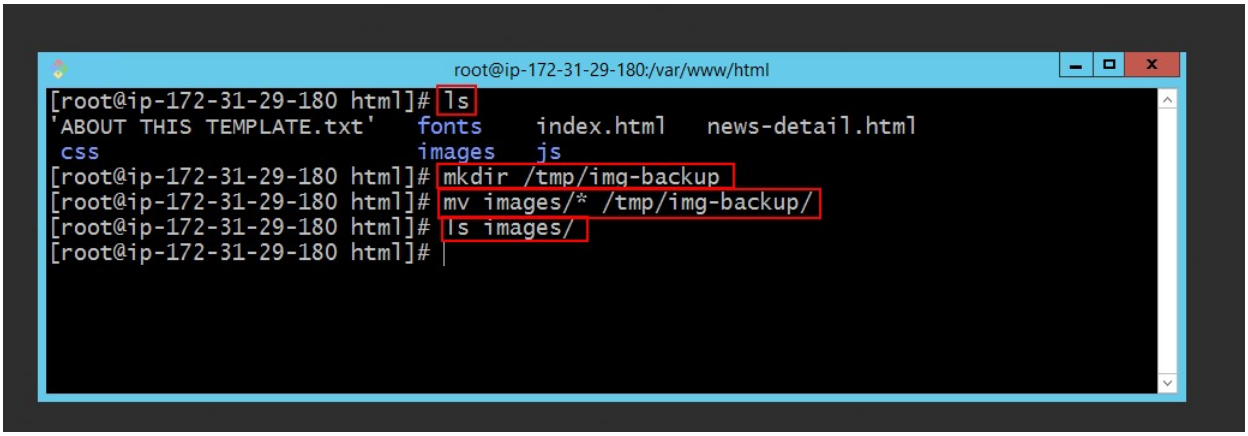
Then, at this stage, click on the **Create Access Point** button.



At this stage, you need to enter the **terminal of your EC2 instance**, and then install the following package to enable access to **EFS**

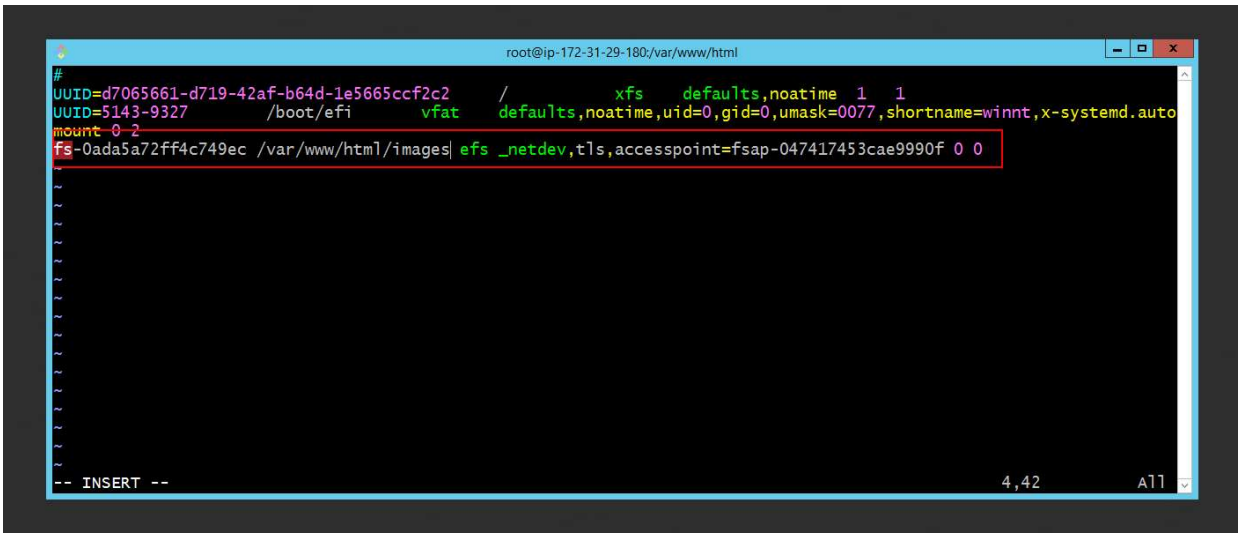


Before mounting **EFS** to the **EC2 instance**, first create a **backup** of the **Images** folder in the web server path

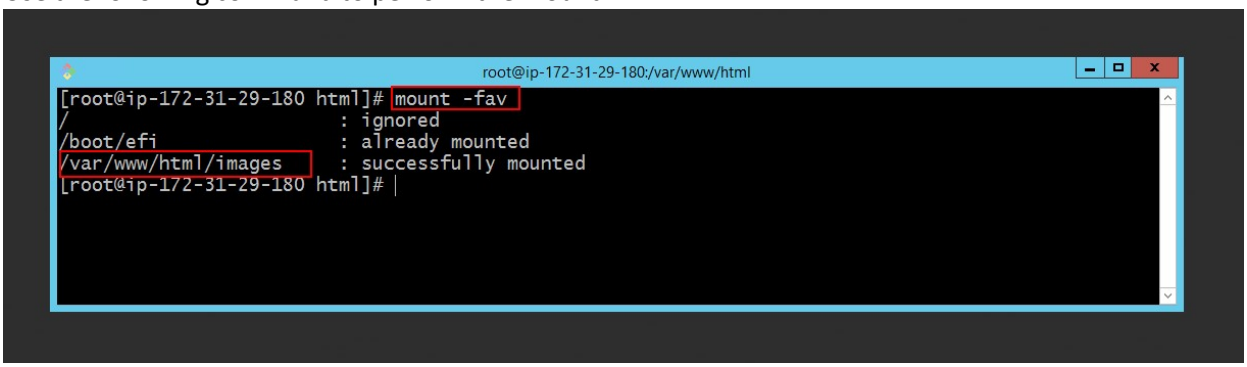


To mount the **EFS**, open the **fstab** file using the following command

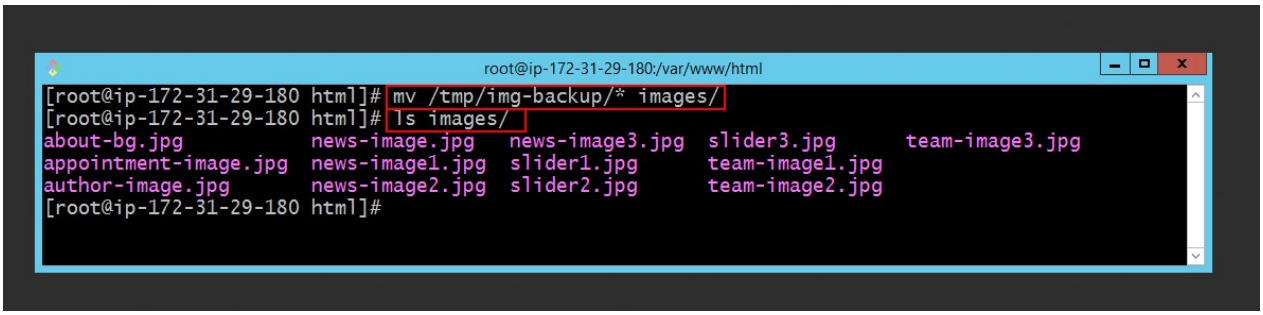
Add the following line to the **fstab** file, then save the file



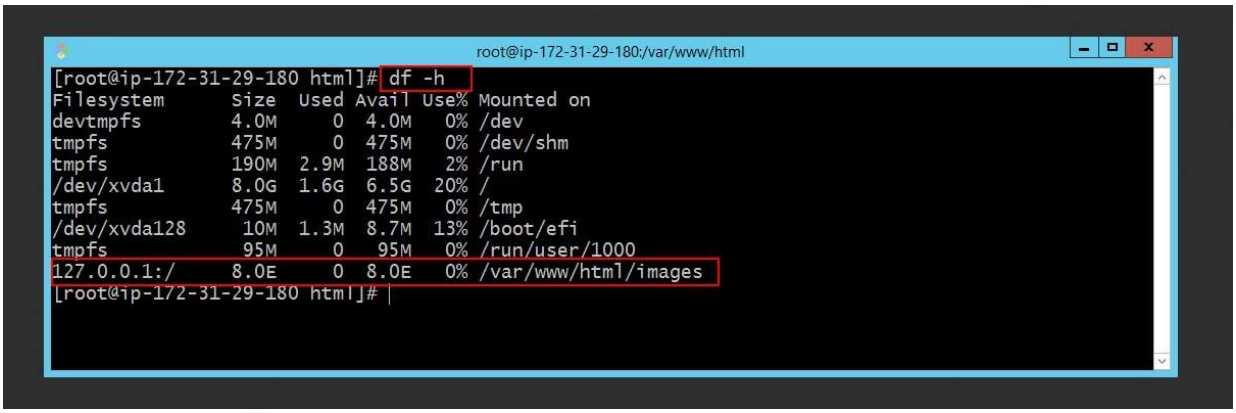
Use the following command to perform the mount



Then, move the files from the **Images** folder backup back to the **web server path** using the following command



Using the following command, you can verify that the **Images** folder is mounted to your EC2 instance and is now located on the **EFS**



Introduction to AWS Auto Scaling

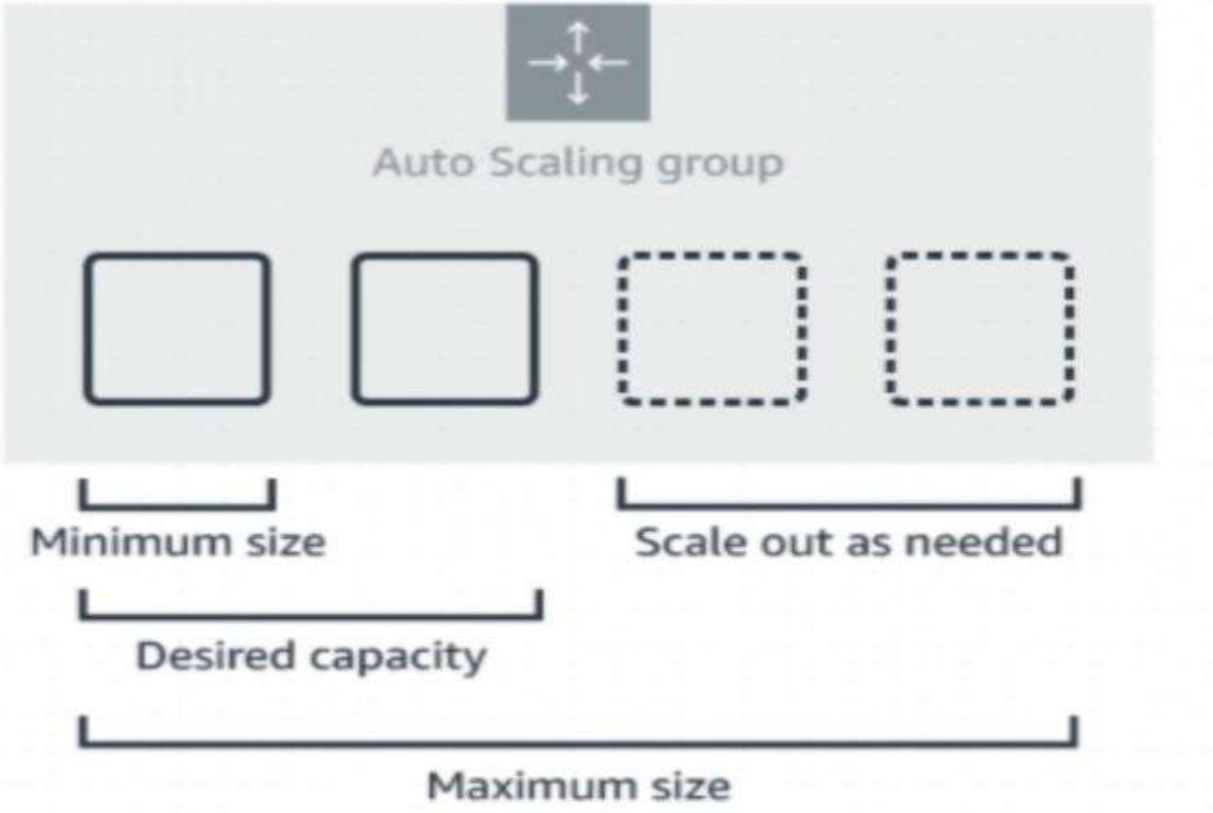
Auto Scaling is a service that can automatically **monitor** and **adjust compute resources** to improve the **performance** of applications hosted on your **AWS** environment.

For example, **Auto Scaling** uses **CloudWatch** to monitor the **CPU usage** of an EC2 instance, and if the CPU usage exceeds the defined **threshold**, it can automatically add one or more instances to the **Auto Scaling Group** to increase performance.

Auto Scaling uses a **Launch Configuration** or **Launch Template** to launch an instance.

A **Scaling Policy** is used to **adjust the capacity**.

An **Auto Scaling Group** contains its own **policies**. For example, a policy can specify that if **CPU usage exceeds 60%**, launch **2 instances**, and if it exceeds **80%**, launch **4 instances**. Alternatively, you can use the **Auto Scaling Group's auto settings** to manage all of this automatically for you.

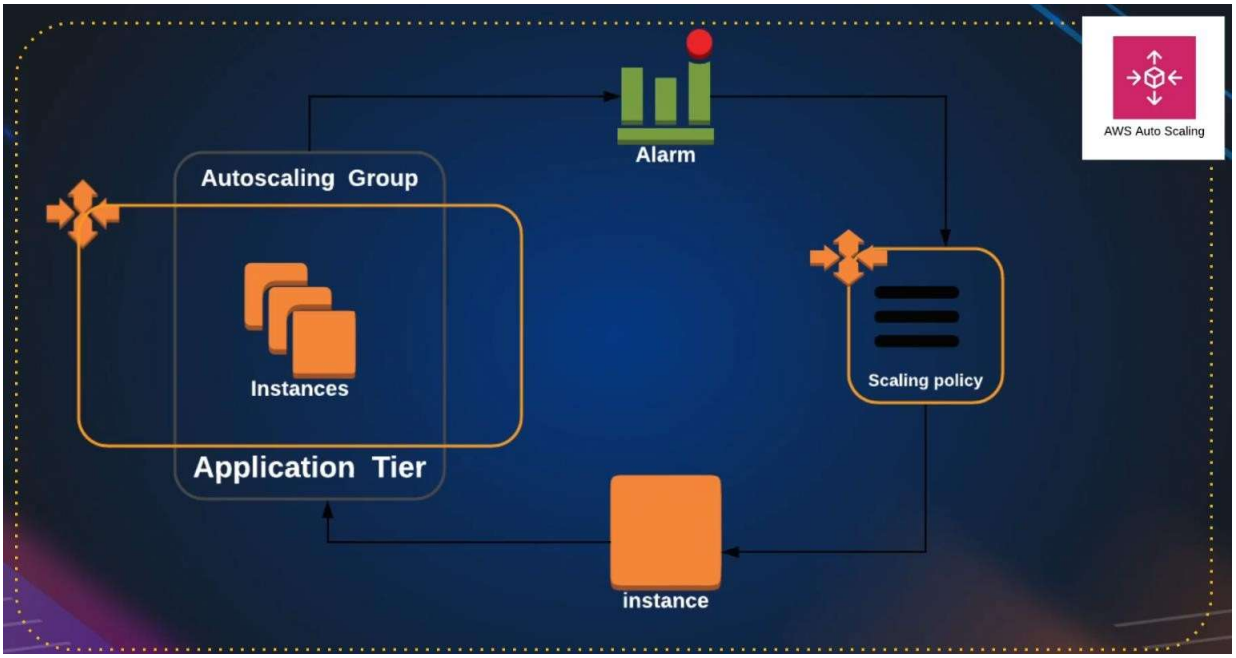


When you create **Auto Scaling**, you can define the **Minimum Size** of the Auto Scaling Group. For example, you can specify **1 instance** as the **minimum instance** in the Auto Scaling Group.

You can also specify the **Desired Capacity**. For example, if you set the **Desired Capacity** to 2 and the **Minimum** to 1, it will launch **2 instances** in the Auto Scaling Group. In other words, there must be **at least one instance** running in the Auto Scaling Group.

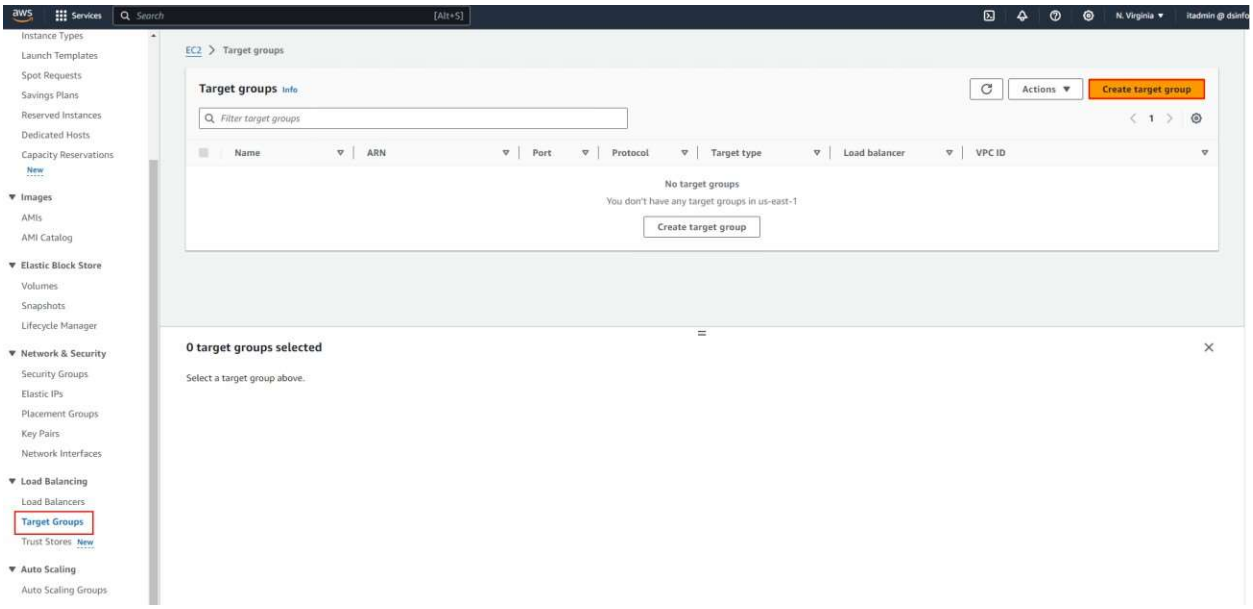
Therefore, when the **Minimum Instance** is set to **1**, during termination, at least **one instance** will remain in the group, and the rest will be terminated. As a result, you'll always have the number of **Desired Instances** running in your **Auto Scaling Group**.

If you set the **Maximum Instance** to **4**, the number of instances added will **not exceed 4**.

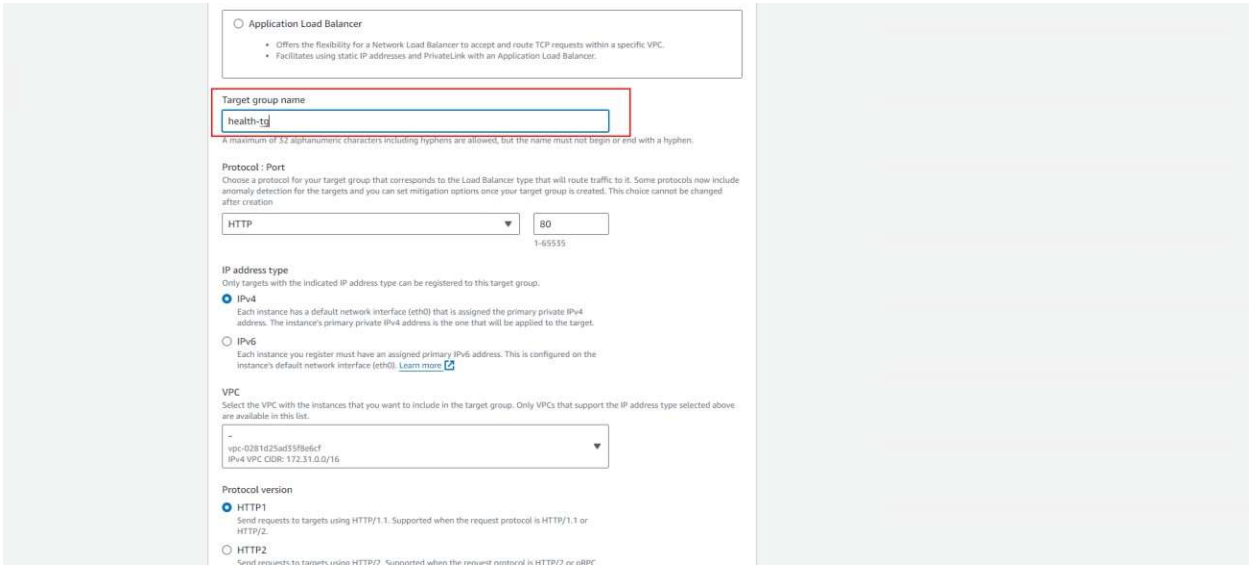


How to Create an AWS Auto Scaling Group

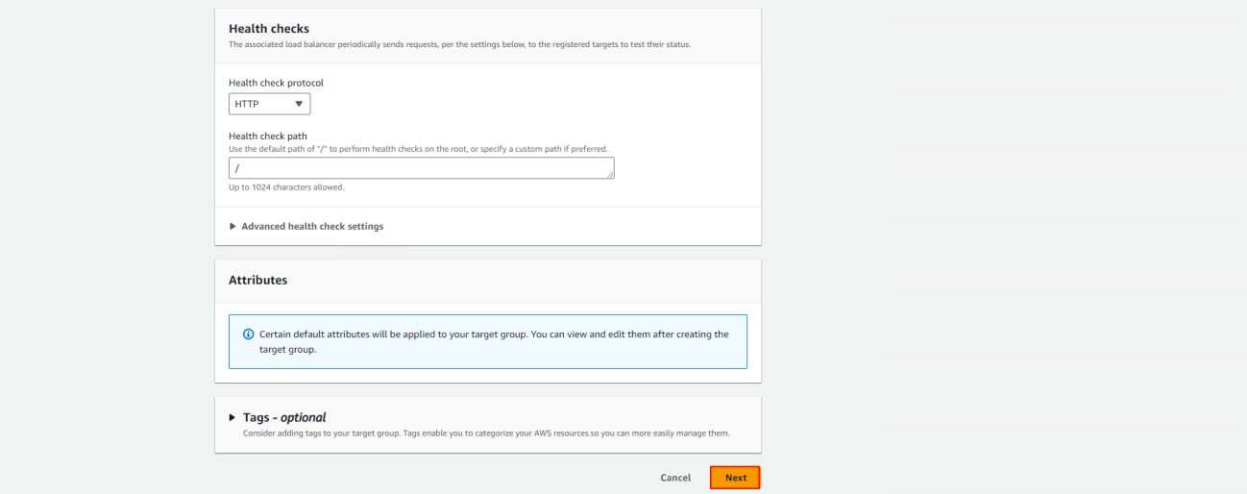
At this stage, first click on the **Target Groups** section, and then click on the **Create Target Group** button.



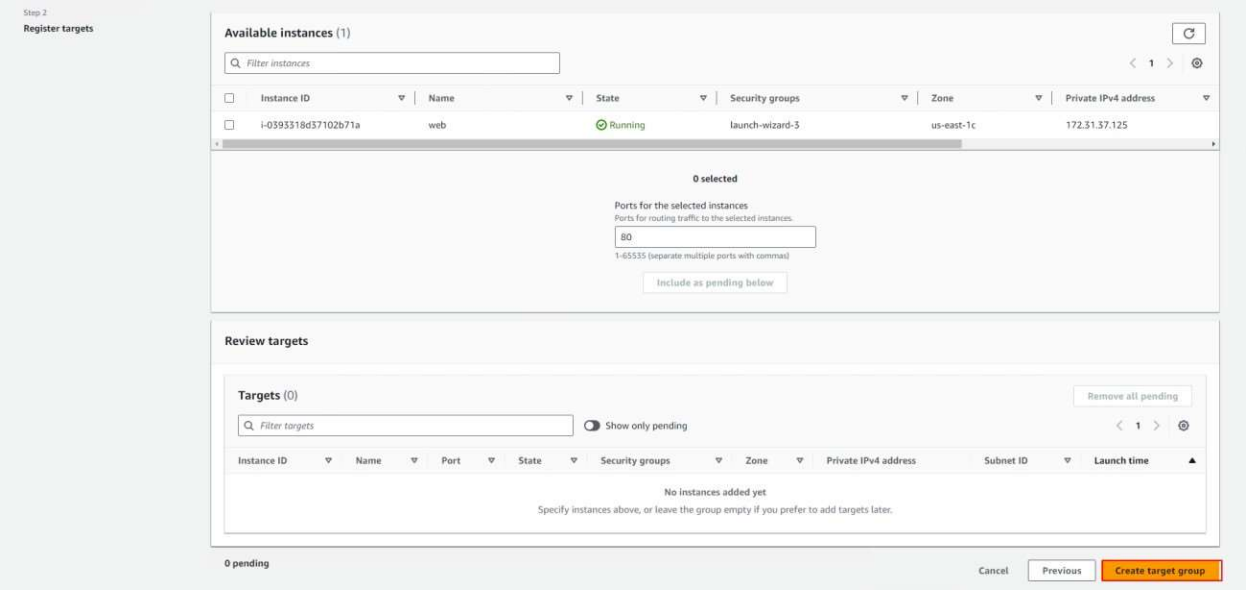
In this section, specify a **name** for the **Target Group**.



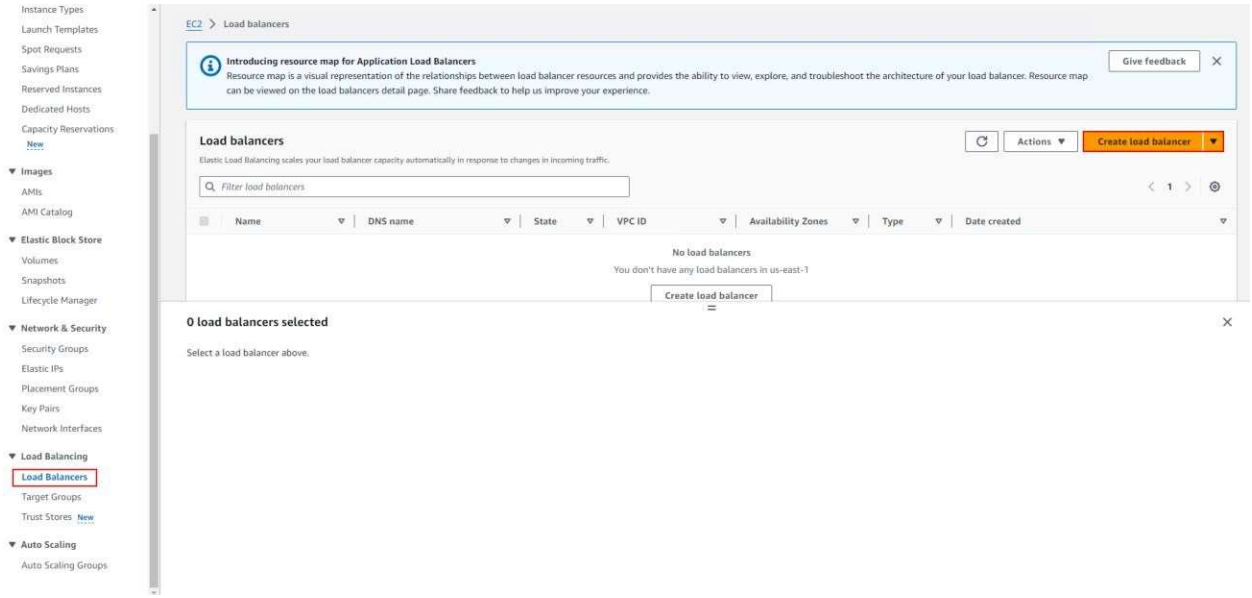
At this stage, click on the **Next** button.



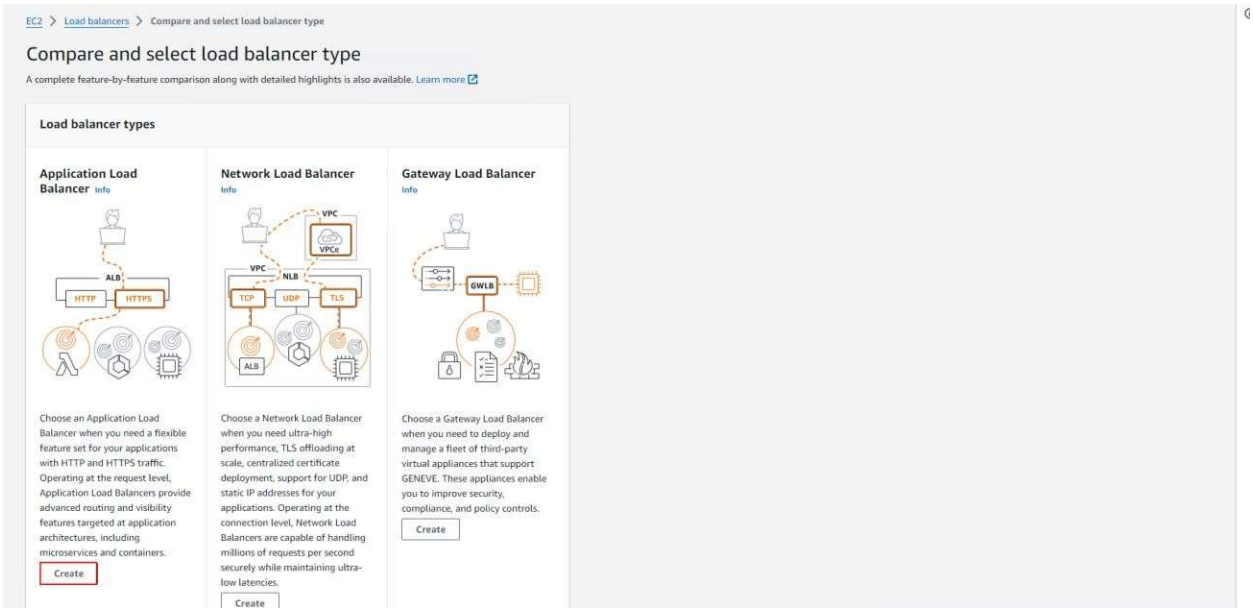
Then, at this stage, click on the **Create Target Group** button.



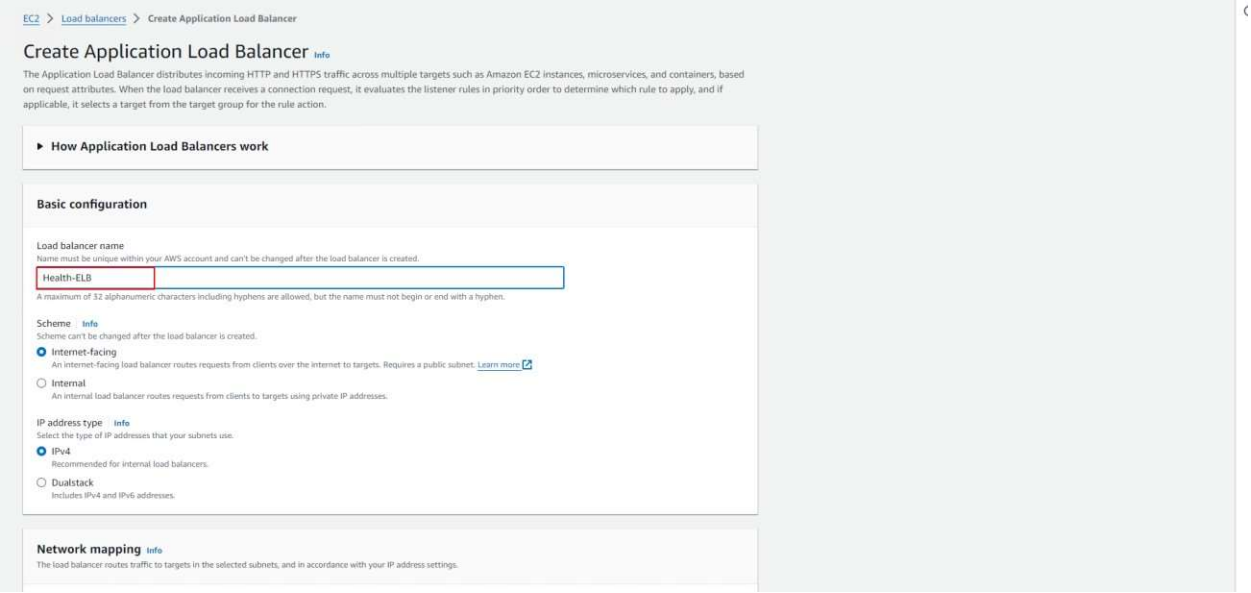
At this stage, click on the **Load Balancers** section, then click on the **Create Load Balancer** button.



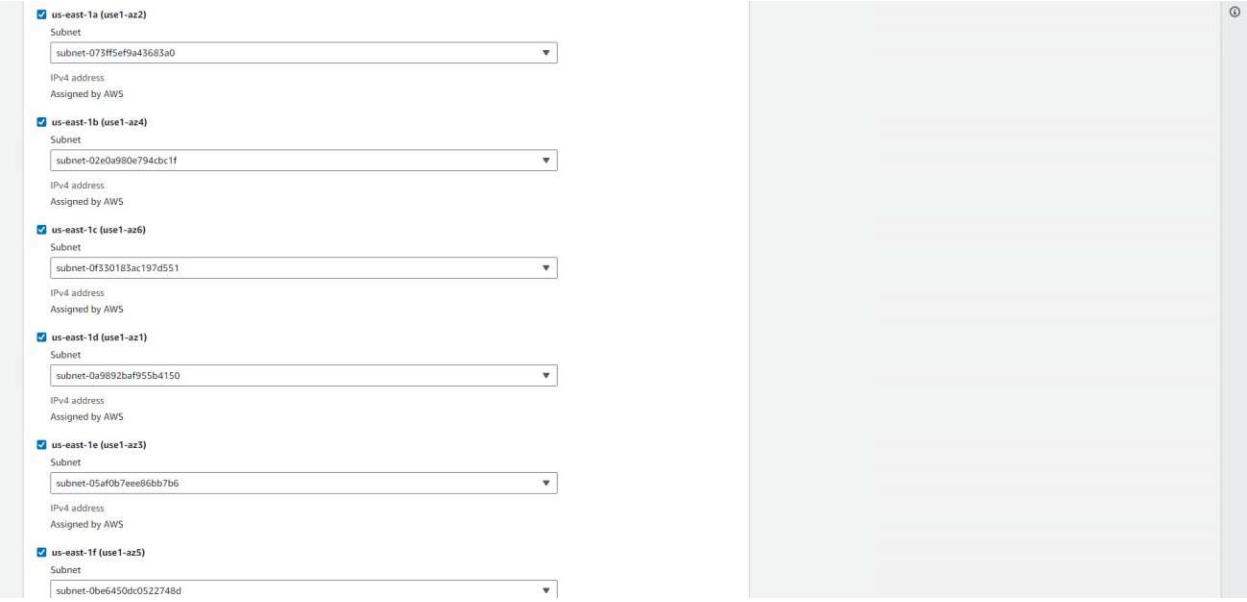
At this stage, select **Application Load Balancer**, and then click the **Create** button.



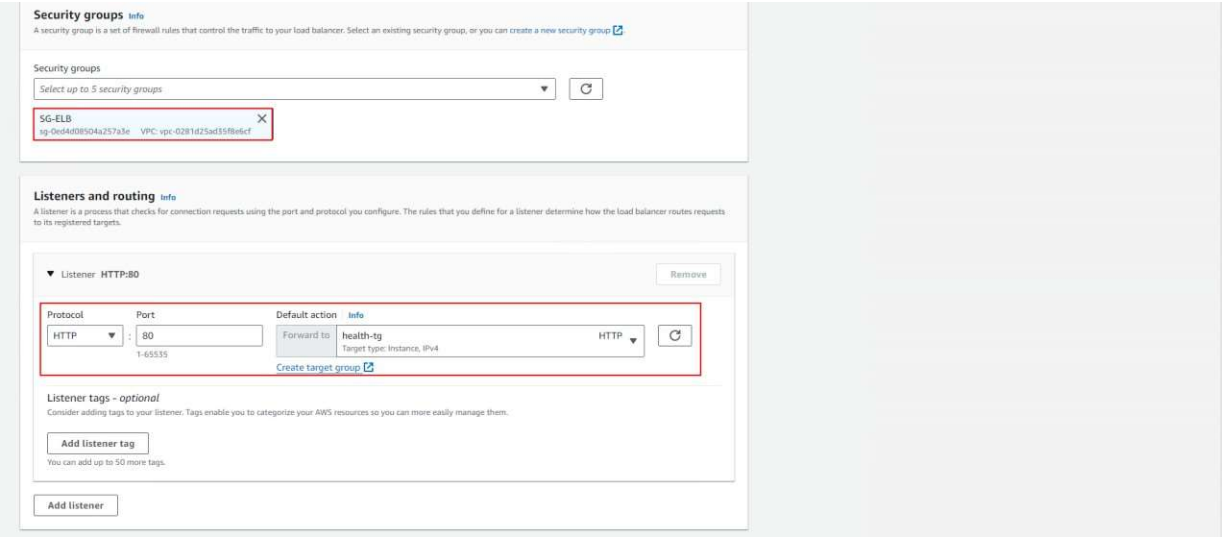
At this stage, specify a **name** for the Load Balancer.



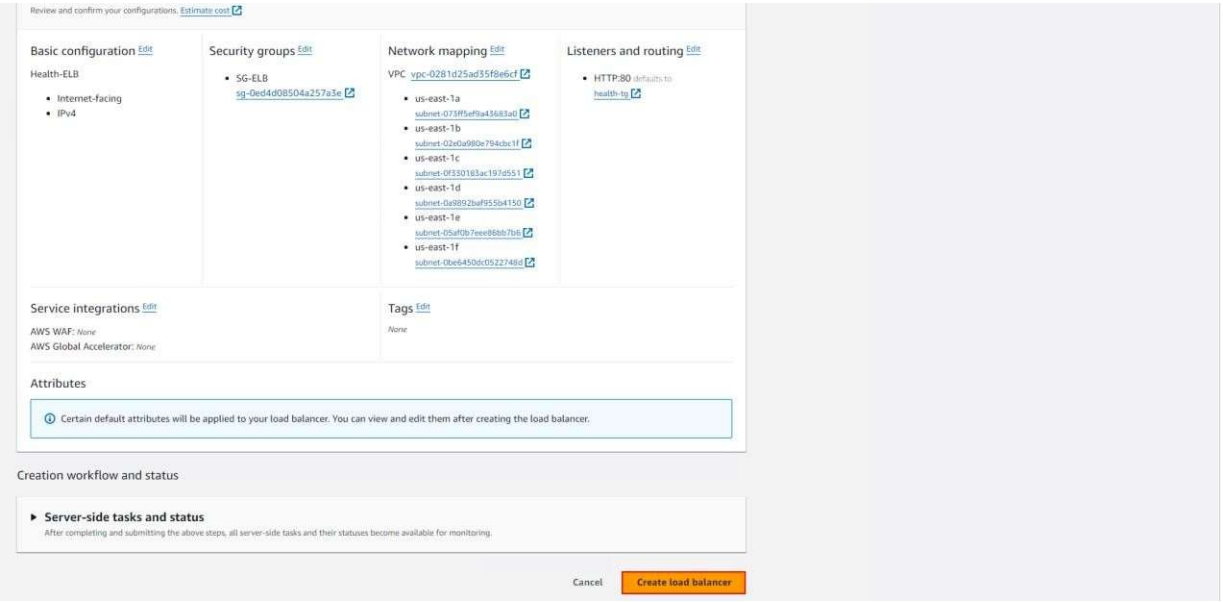
At this stage, select **all Availability Zones** to increase **availability**.



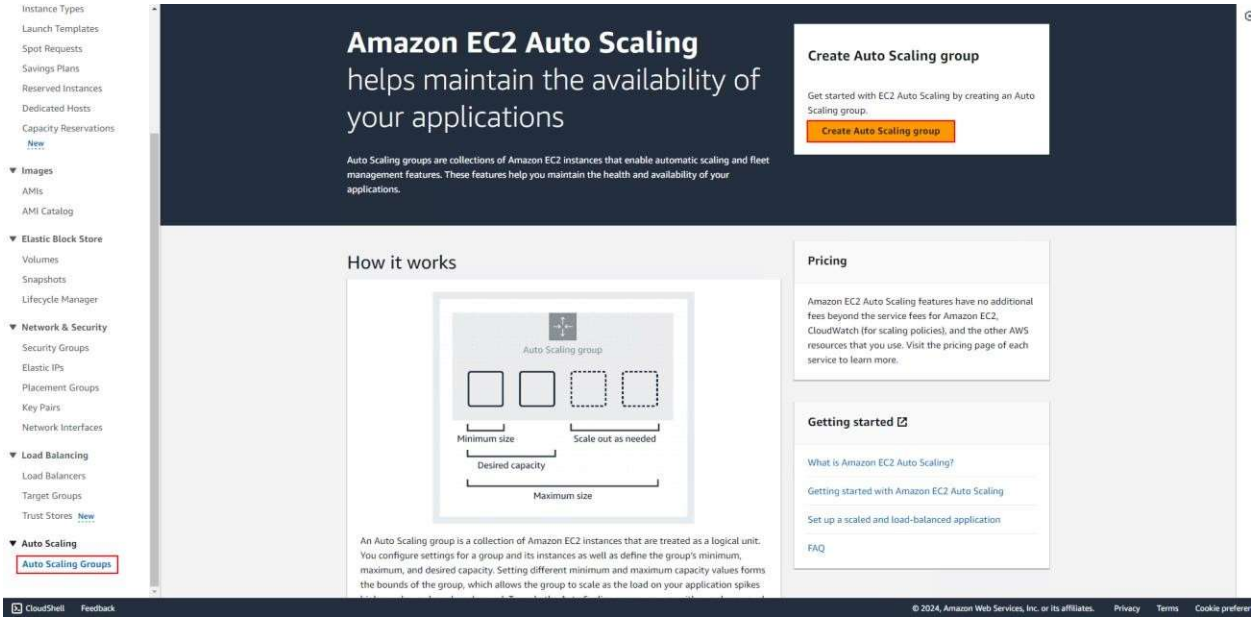
At this stage, select the **SG-ELB** as the **Security Group**, and in the **Listener** section, set the **Default Action** to the **Target Group** you defined in the previous step.



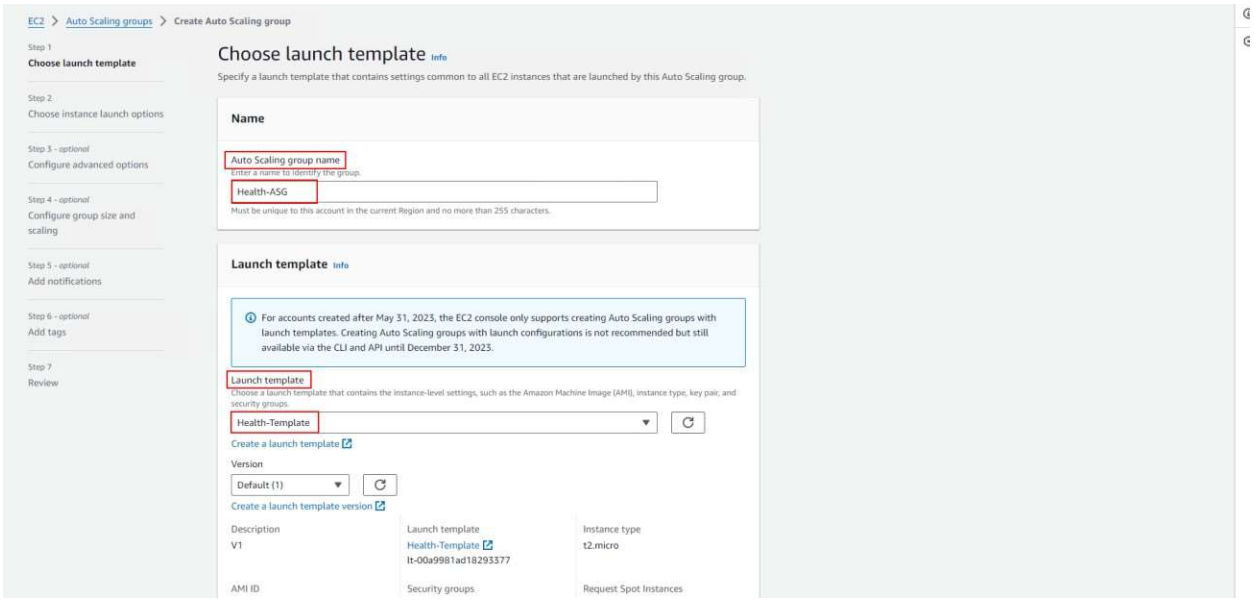
And then, at this stage, click on the **Create Load Balancer** button.



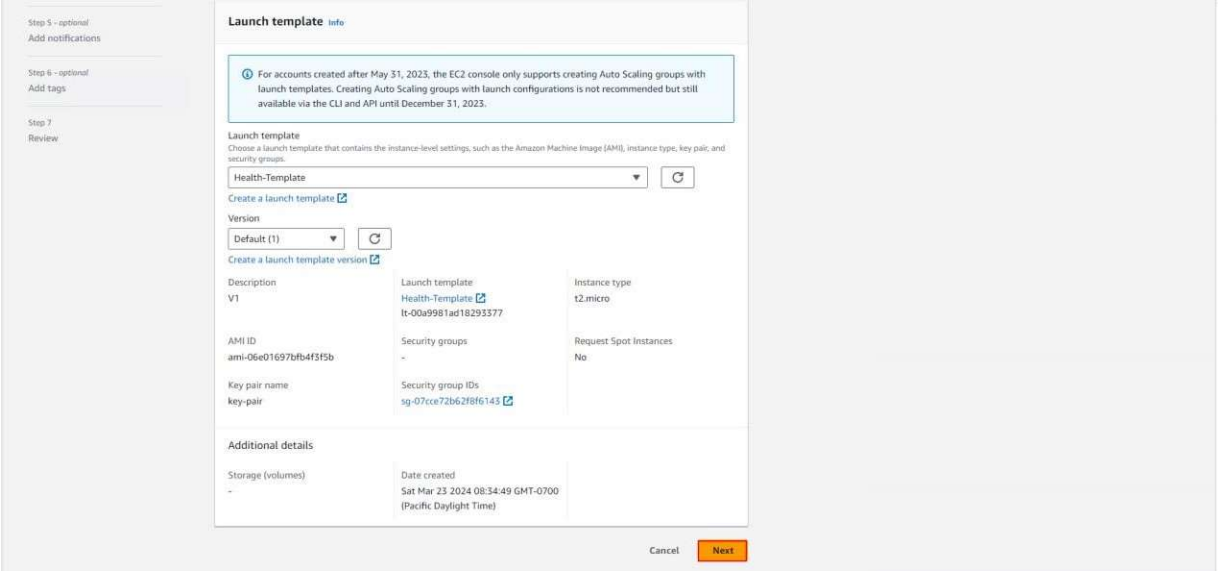
At this stage, click on the **Auto Scaling Groups** section, and then click on the **Create Auto Scaling Group** button.



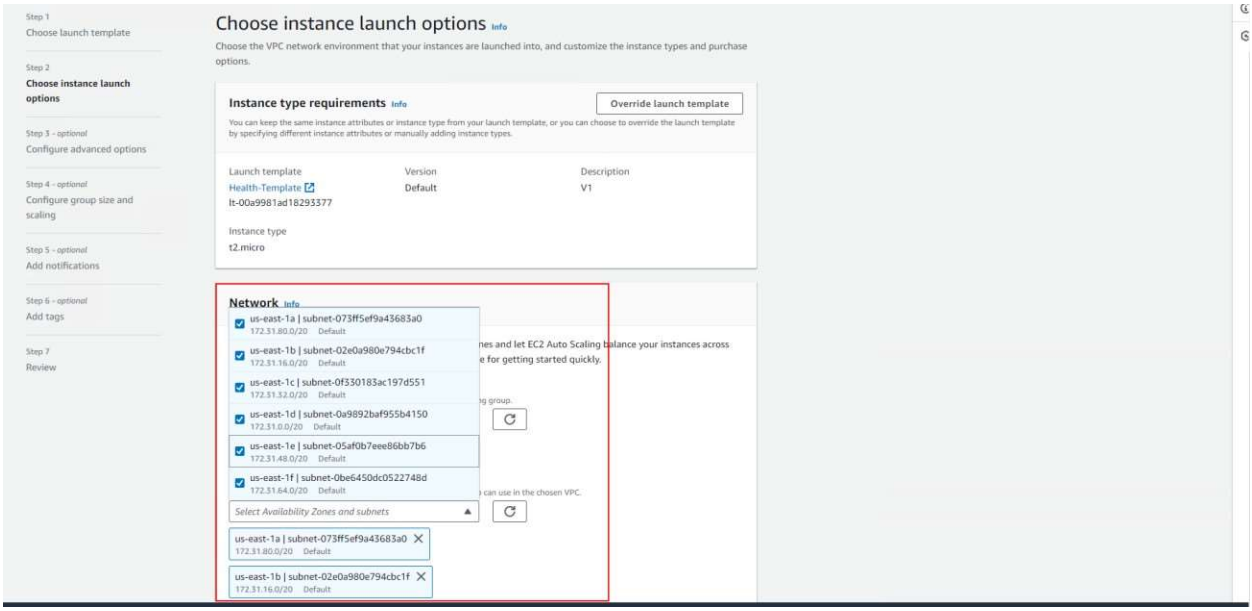
At this stage, specify a **name** for the Auto Scaling Group, and in the **Launch Template** section, select the **template** you created earlier.



At this stage, click on the **Next** button.

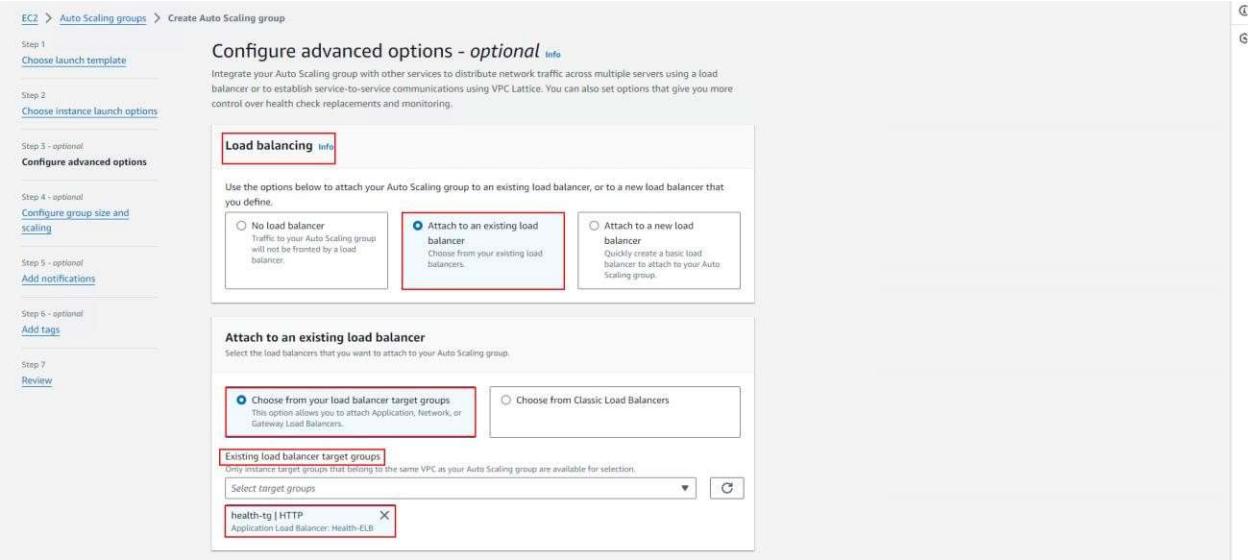


At this stage, select **all Availability Zones** to enhance **availability**.

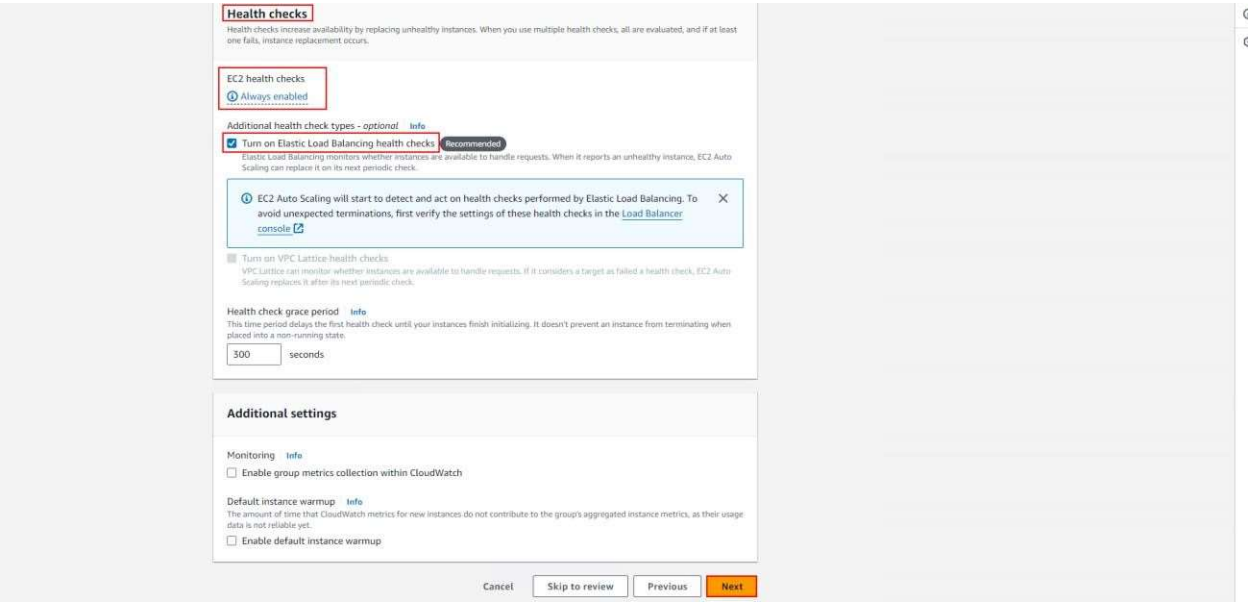


Then, at this stage, click on the **Next** button.

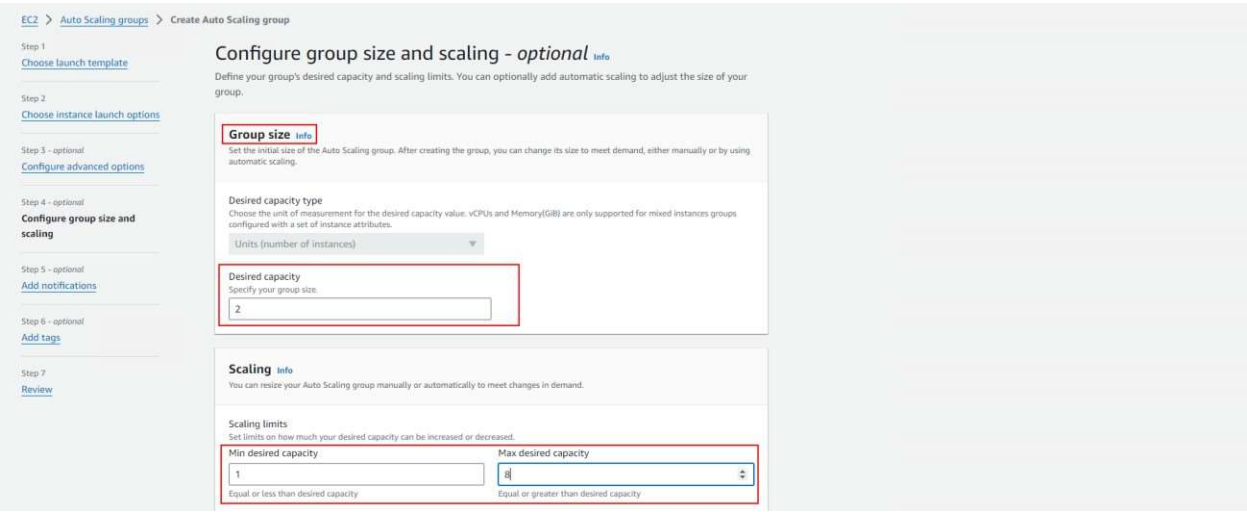
At this stage, in the **Load Balancing** section, select the option **Attach to an existing load balancer**, and then specify the **Target Group**.



At this stage, select the option shown below, and then click on the **Next** button.

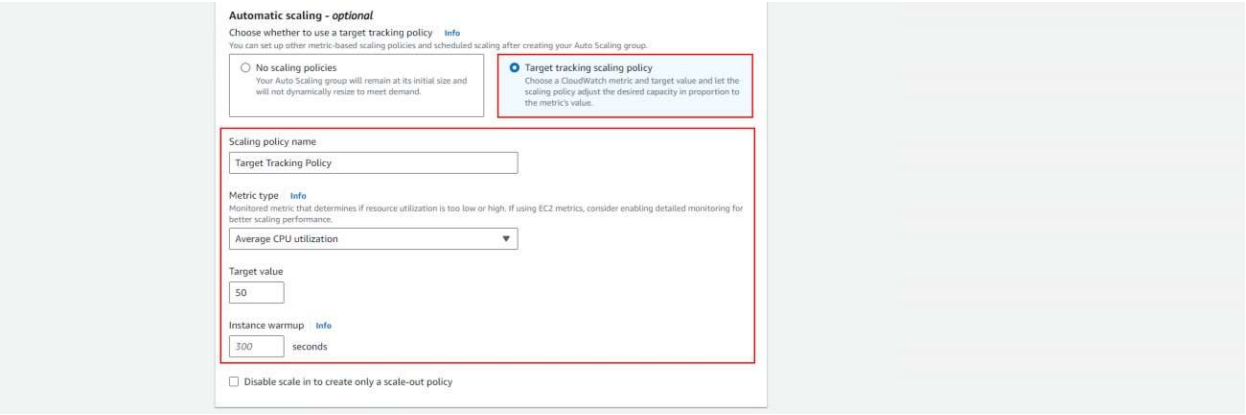


At this stage, you need to configure the **Group Size** settings, as shown in the image below.

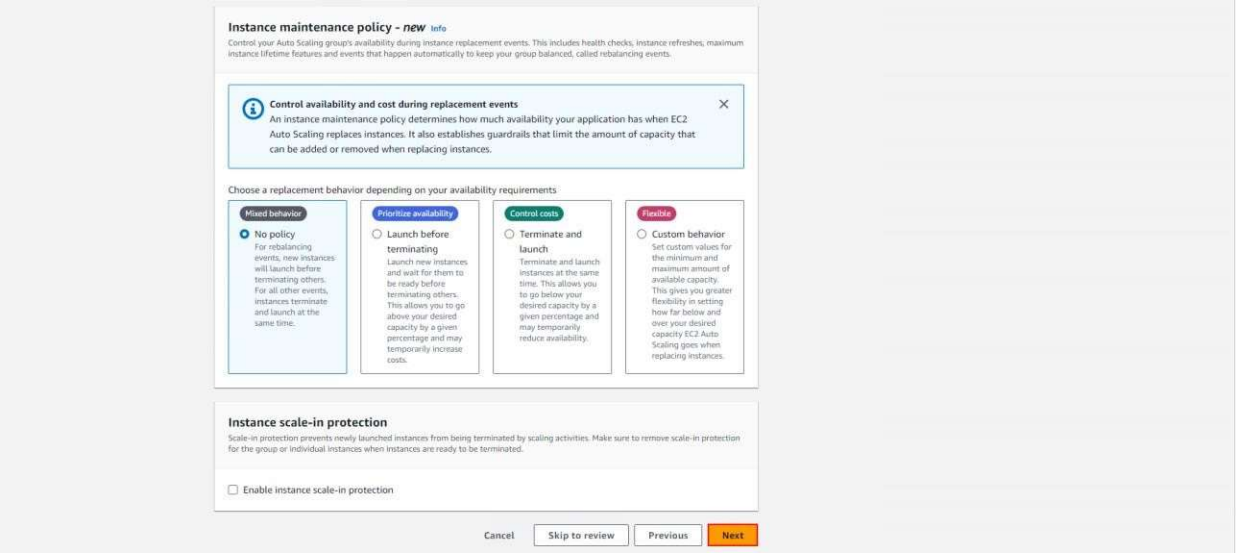


At this stage, select the **Target Tracking Scaling Policy** option, then set the **Metric Type** to **Average CPU Utilization**, and set the **Target Value** to **50%**.
This means that if the average CPU utilization exceeds 50%, Auto Scaling will increase the

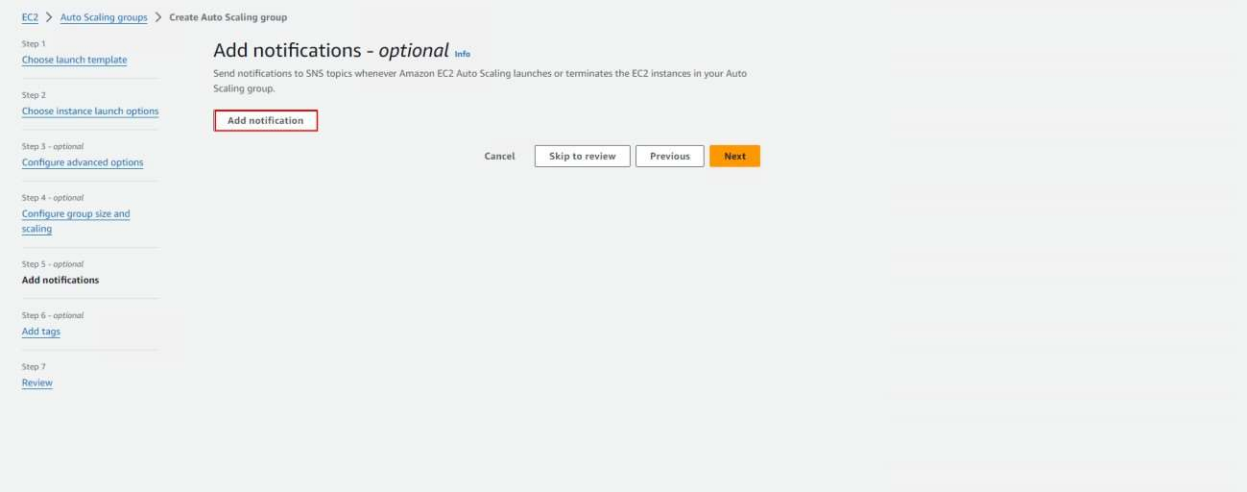
number of instances up to the **Maximum Value**, and if it drops to 50% or below, it will reduce the instances back to the **Desired Value**.



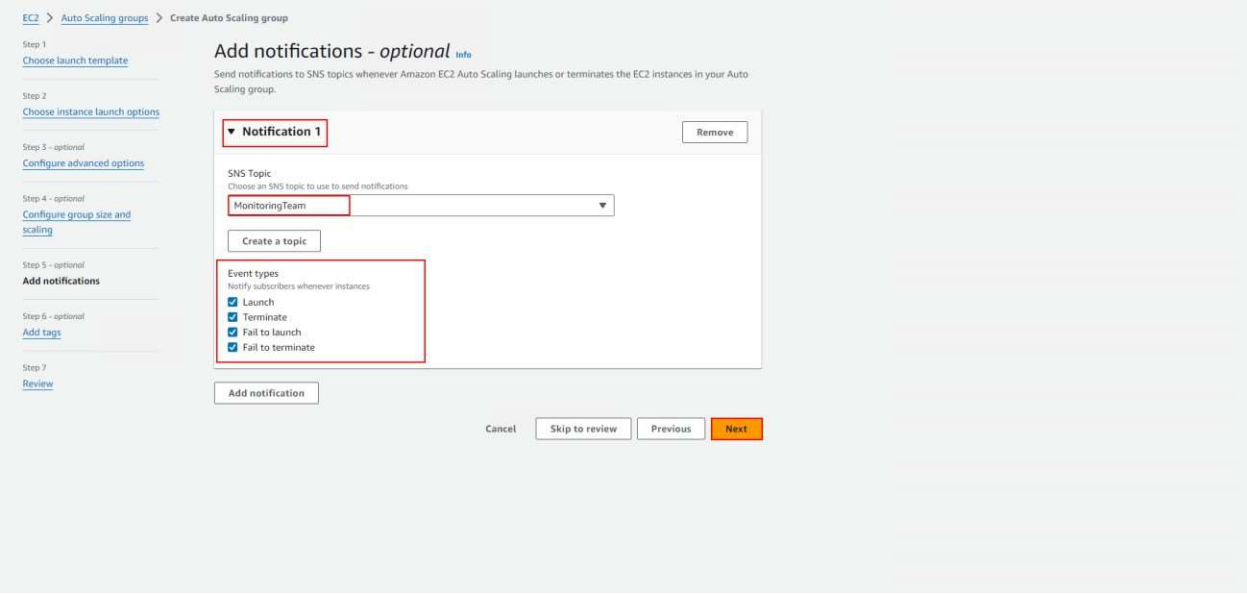
At this stage, click on the **Next** button.



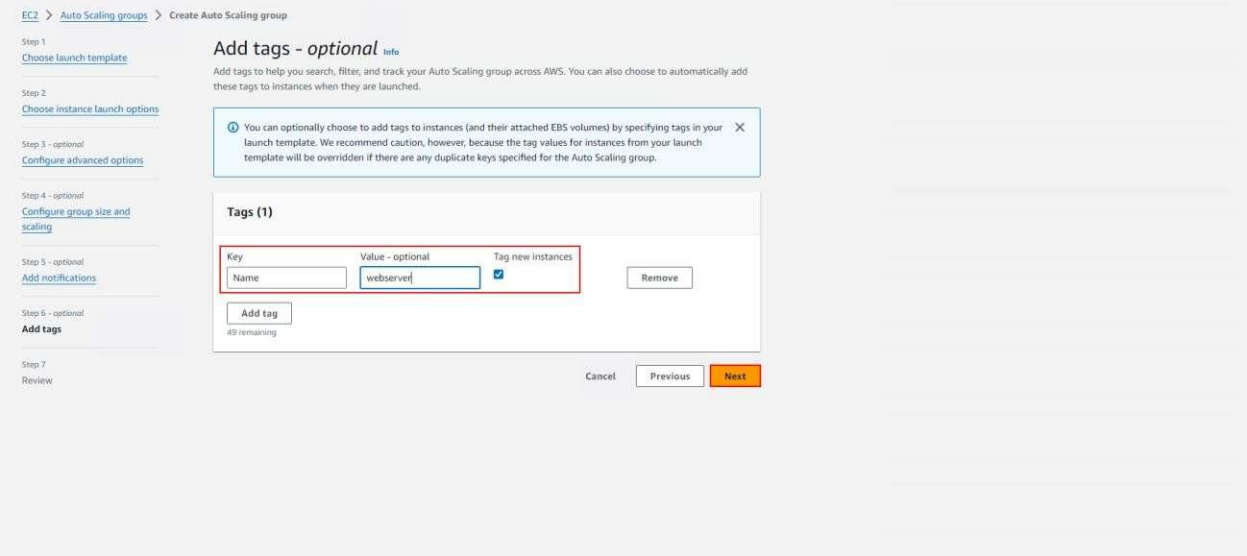
At this stage, click on the **Add Notification** button.



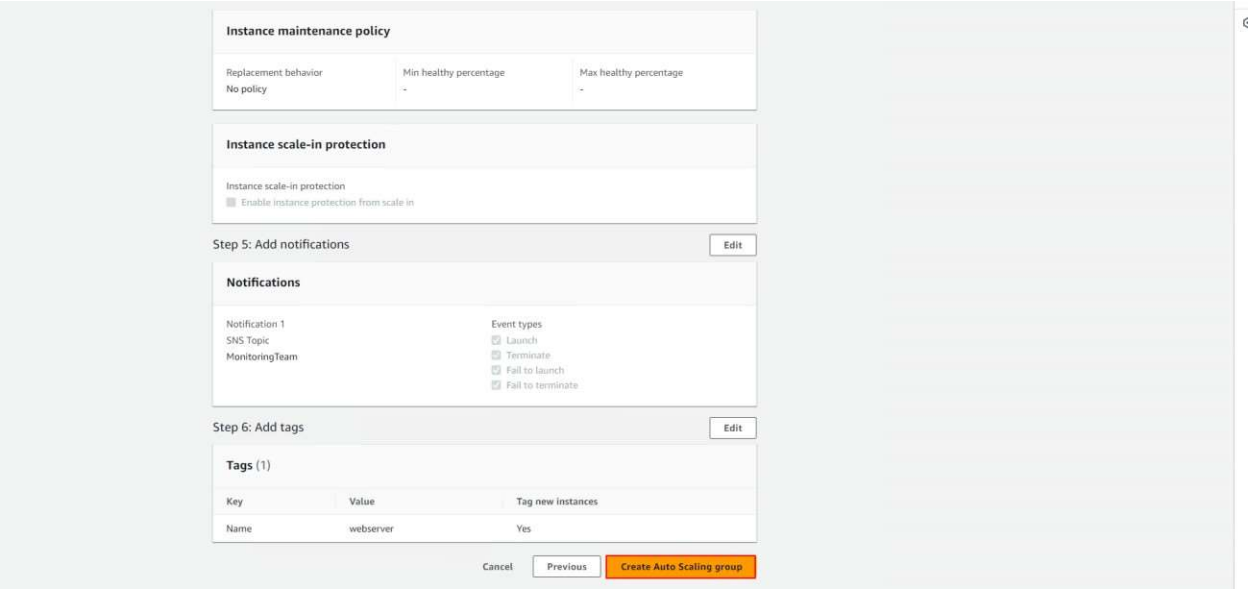
At this stage, select the **SNS Topic** and **Event Type**, then click on the **Next** button.



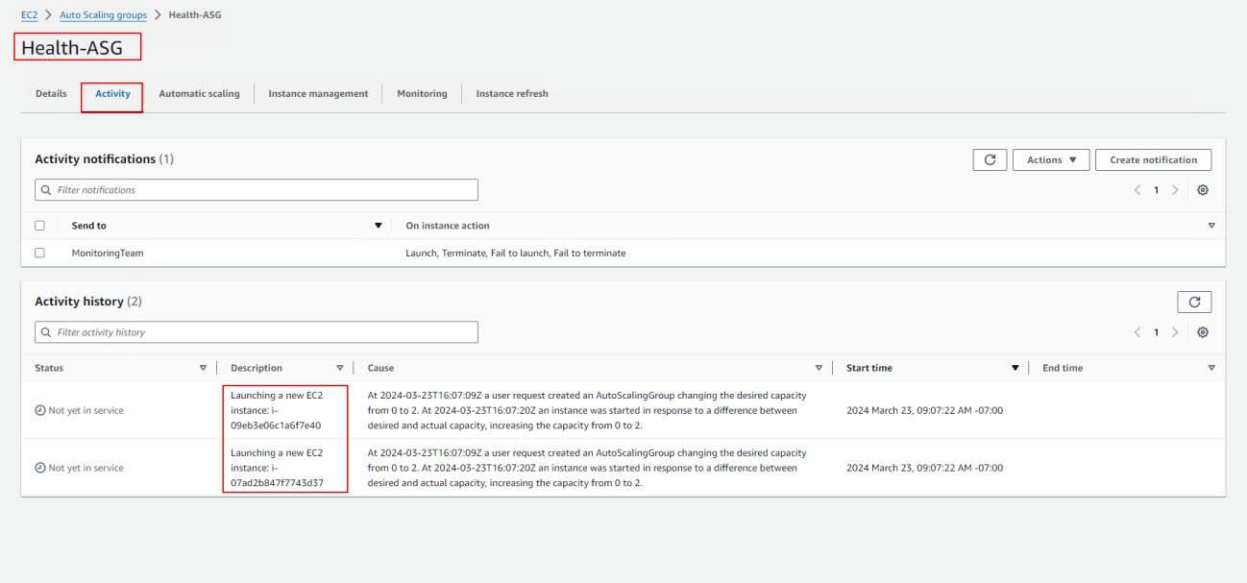
At this stage, define a **Tag** for the Auto Scaling Group, and then click on the **Next** button.



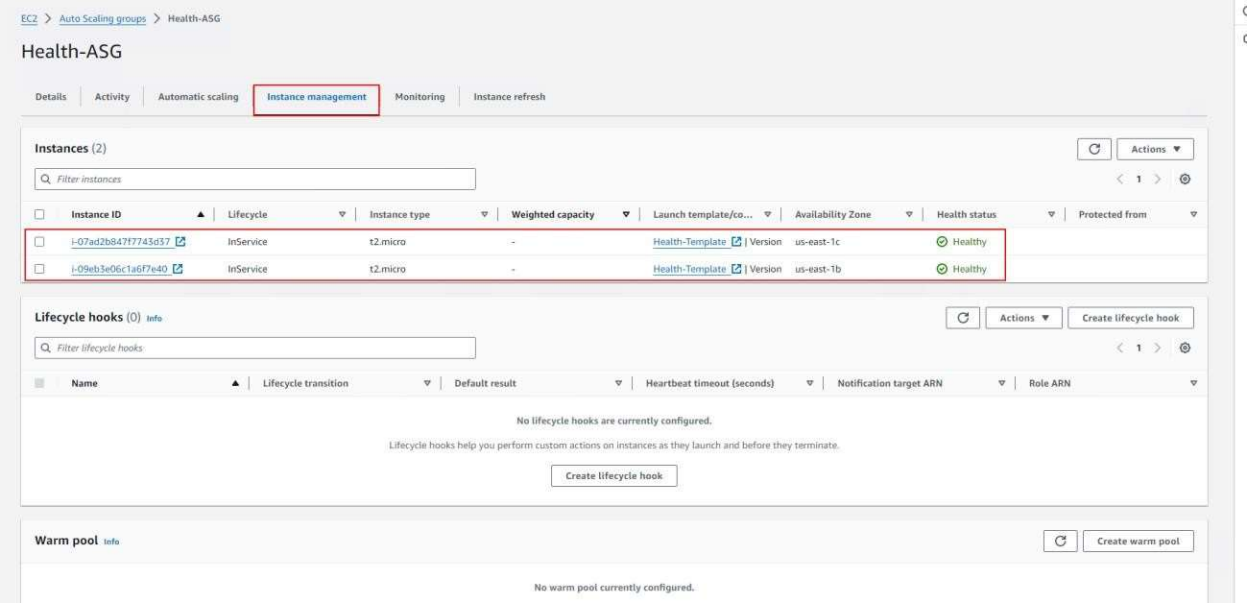
At this stage, click on the **Create Auto Scaling Group** button.



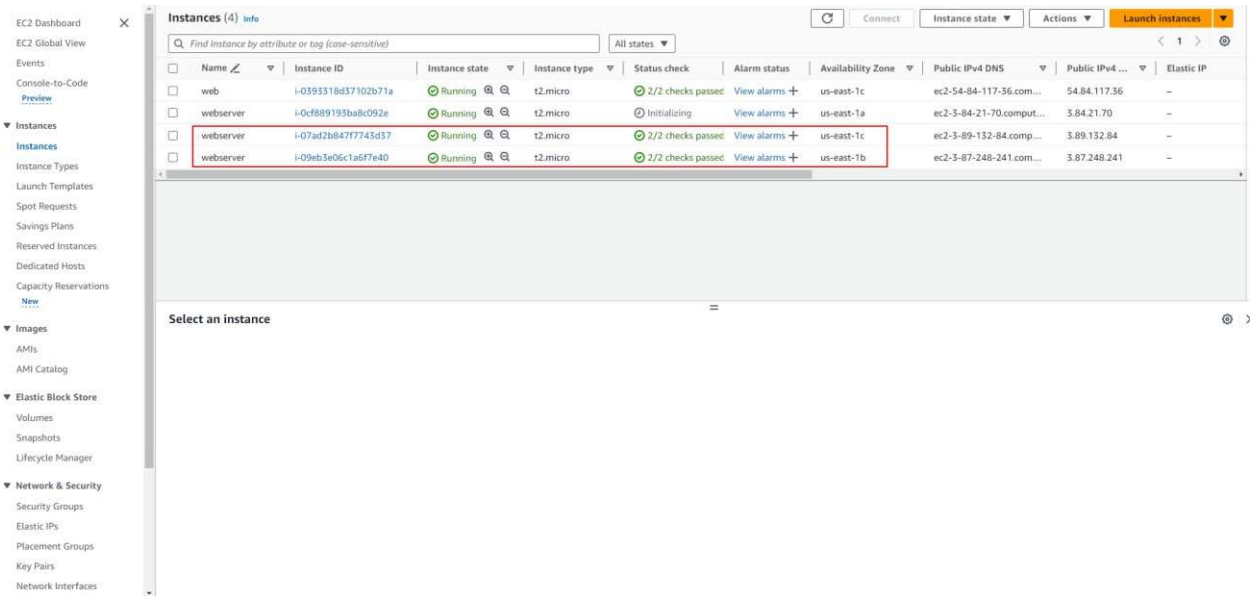
In the **Activity** section of the **Auto Scaling Group**, you can see that **two instances** have been created.



In the **Instance Management** section of the **Auto Scaling Group**, you can view their **Health Status**.



And in the **Instances** section, you can view the **instances created by Auto Scaling**.



Introduction to Amazon S3

The **S3 service**, short for **Simple Storage Service**, is a **storage service** that is **very easy to use**.

The **S3 service** is one of the **most popular and oldest services** offered by **AWS**.

The **S3 service** is an **internet-based storage service**. You can use **S3** to **store** and **retrieve** data from **anywhere**, at **any time**.

You can think of the **S3 service** as being similar to **Google Drive** or **Dropbox**, or even **more powerful** than them.

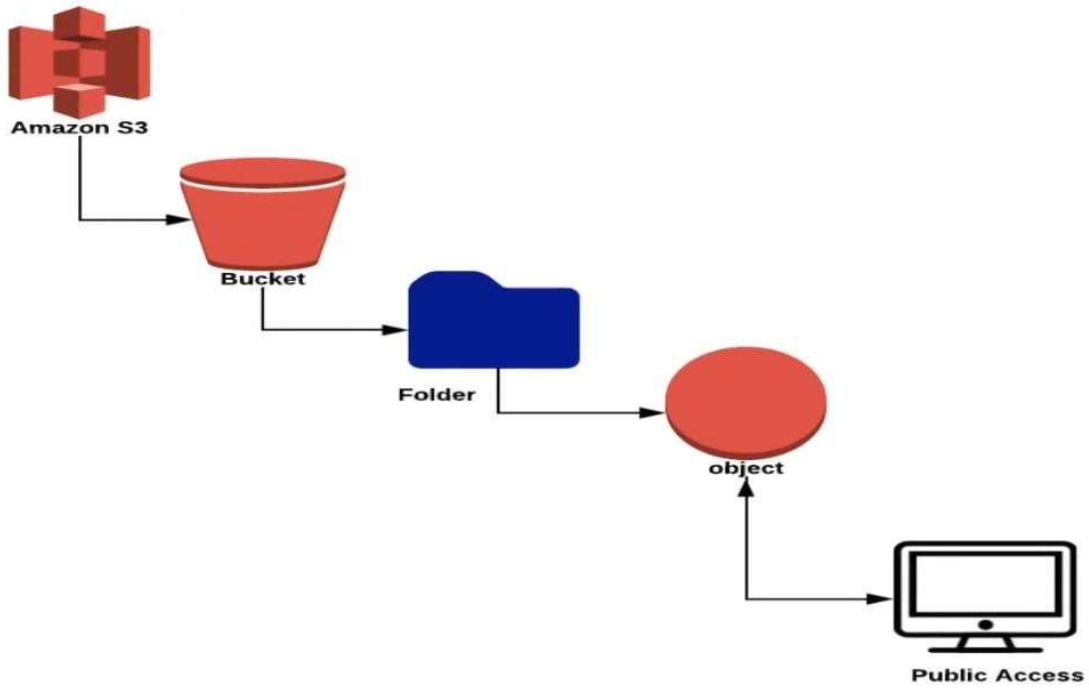
S3 is an **Object Storage** service where you can upload any type of file, such as **documents**, **pictures**, or **videos**, and access them from **anywhere**.

When you upload any data to your **S3 bucket**, the **bucket** acts as a **top-level storage** container—similar to a **folder** in S3.
Your data is **replicated across multiple S3 systems**, and there are **no limits** on storage.

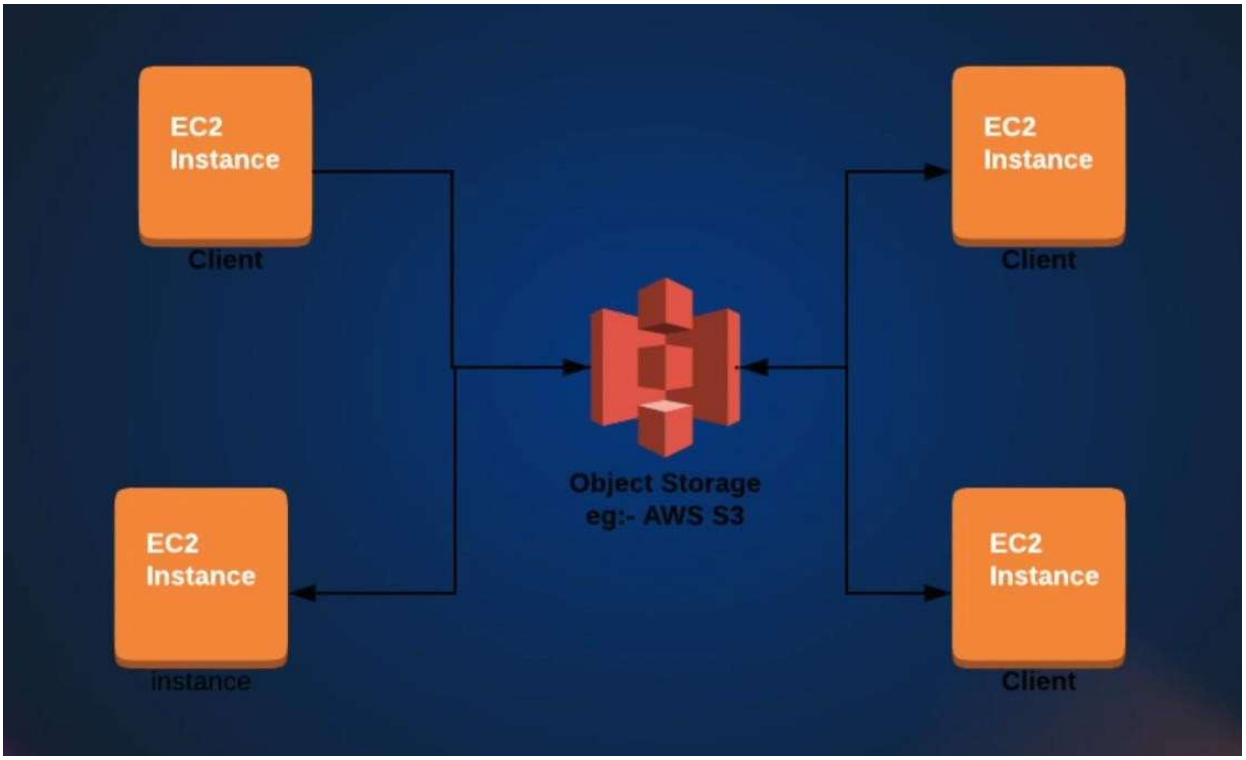
The **S3 service** offers **unlimited storage**, allowing you to store **as much data as you want**.

The data stored in **S3** is referred to as an **Object**, and the storage container itself is called a **Bucket**.

The **bucket name** must be **unique**.



One common use of an **S3 bucket** is when you want to **store data** from your **EC2 instances** on **S3**.



Types of S3 Storage Classes:

1. **S3 Standard**
   - Designed for **frequent access**
   - High durability and availability
   - Suitable for general-purpose storage
2. **S3 Intelligent-Tiering**
   - Automatically moves data between two tiers (frequent and infrequent access)
   - Optimizes costs without performance impact
3. **S3 Standard-IA (Infrequent Access)**
   - For data that is accessed **less frequently**, but still requires rapid access
   - Lower storage cost, higher retrieval cost
4. **S3 One Zone-IA**
   - Similar to Standard-IA but stored in a **single Availability Zone**
   - Lower cost, but less resilient to AZ failure
5. **S3 Glacier**
   - For **archival** and long-term backups

    - Retrieval time from **minutes to hours**
6. **S3 Glacier Deep Archive**
    - **Lowest-cost** storage class
    - Retrieval time can take **up to 12 hours**
    - Best for long-term **compliance** and **archiving**

Each class is optimized for a different **use case** and **access pattern**, allowing you to choose based on cost, retrieval speed, and availability needs.

Introduction to Storage Lifecycle Policy in S3

An **S3 Lifecycle Policy** is a set of rules that **automates the transition** of objects between **storage classes** or **deletes them** after a certain period of time. This helps you **optimize costs** and manage data efficiently.

Key Capabilities:

- **Transition Rules**
  Automatically move objects from one storage class to another (e.g., from Standard to Glacier) after a specific number of days.
- **Expiration Rules**
  Automatically delete objects after a defined period of time (e.g., delete logs older than 90 days).

Common Use Cases:

- Move **infrequently accessed data** to **S3 Standard-IA** after 30 days.
- Archive data to **Glacier** or **Deep Archive** after 90 or 180 days.
- **Delete old backups or log files** after 1 year to save on storage costs.

Lifecycle policies are defined at the **bucket level**, and you can apply different rules based on **prefixes** (folders) or **tags**.

Introduction to S3 Charges (S3 Pricing)

Amazon S3 charges are based on multiple factors, and understanding them helps you manage costs effectively.

Key Factors That Affect S3 Pricing:

1. **Storage Used**
   - You are charged based on the total amount of data stored per month, depending on the **storage class** (e.g., Standard, Glacier).
2. **Number of Requests**
   - Charges apply for **GET, PUT, COPY, POST, DELETE** requests.
   - More requests = higher cost, especially in high-traffic applications.
3. **Data Transfer**
   - **Data transferred out** of S3 to the internet is billed (first 1 GB per month is free).
   - **Data transfer within the same region** (e.g., S3 to EC2) is often free.
4. **Management Features**
   - Features like **replication**, **object tagging**, **lifecycle policies**, and **inventory** may incur additional costs.
5. **Early Deletion Fees**
   - Some classes like **Glacier** and **Standard-IA** have minimum storage durations. Deleting files early may lead to extra charges.

Tips to Reduce S3 Costs:

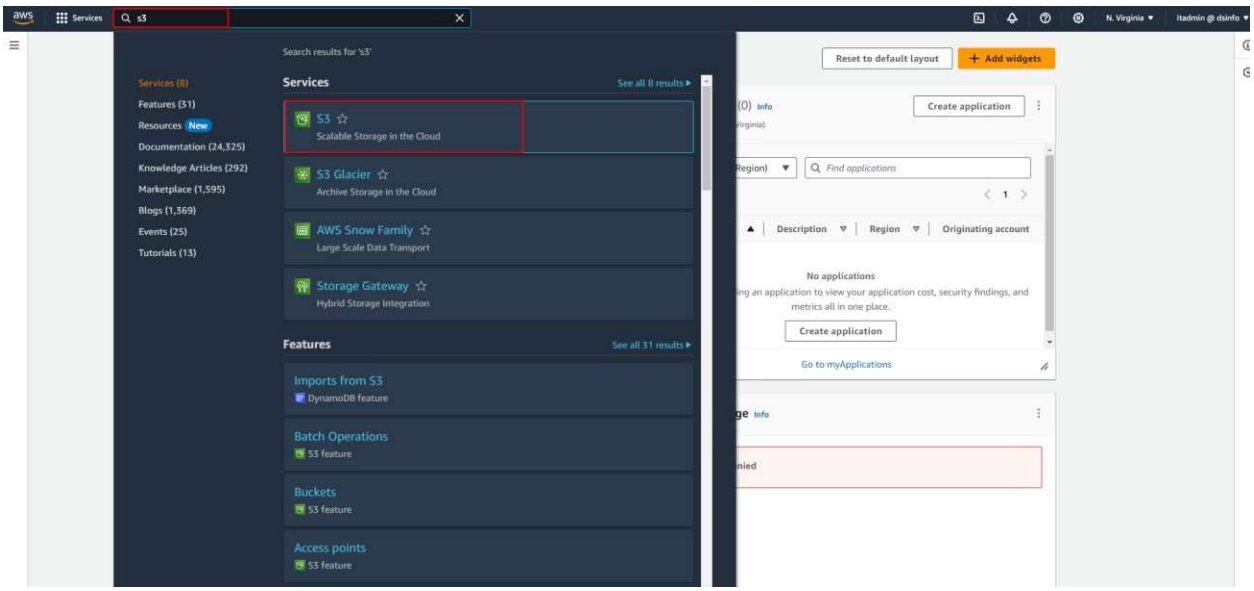
- Use **lifecycle policies** to move infrequently accessed data to lower-cost storage classes.

- Delete unused or outdated objects regularly.
- Minimize unnecessary read/write operations.

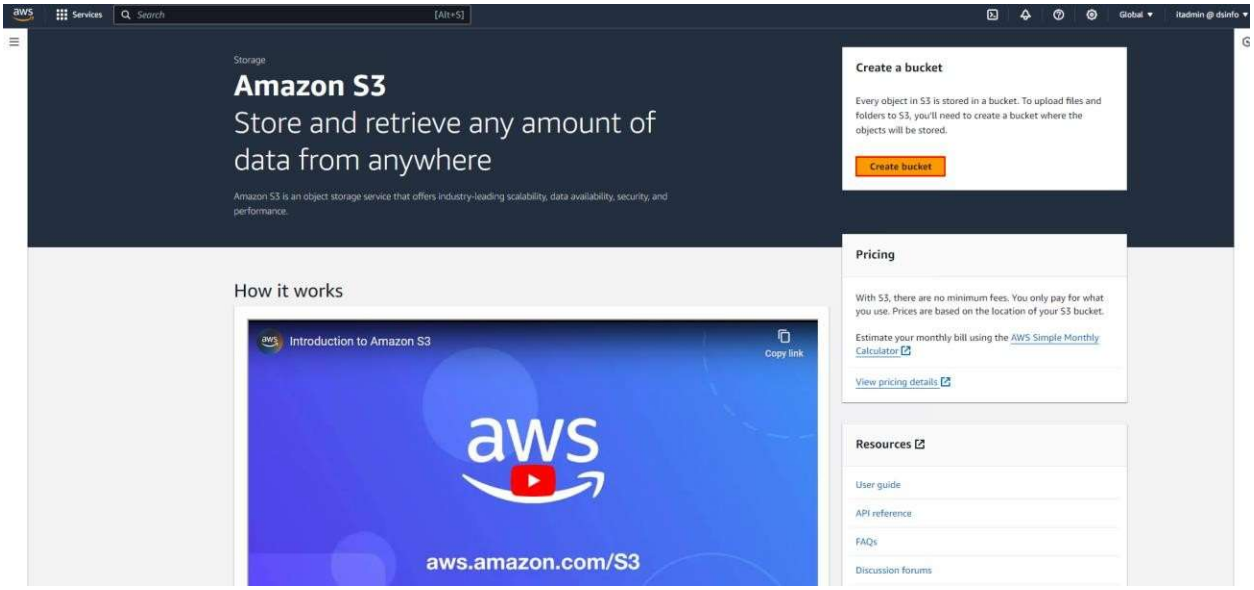
Always check the **AWS Pricing Calculator** for accurate and updated cost estimates.

How to Use AWS S3

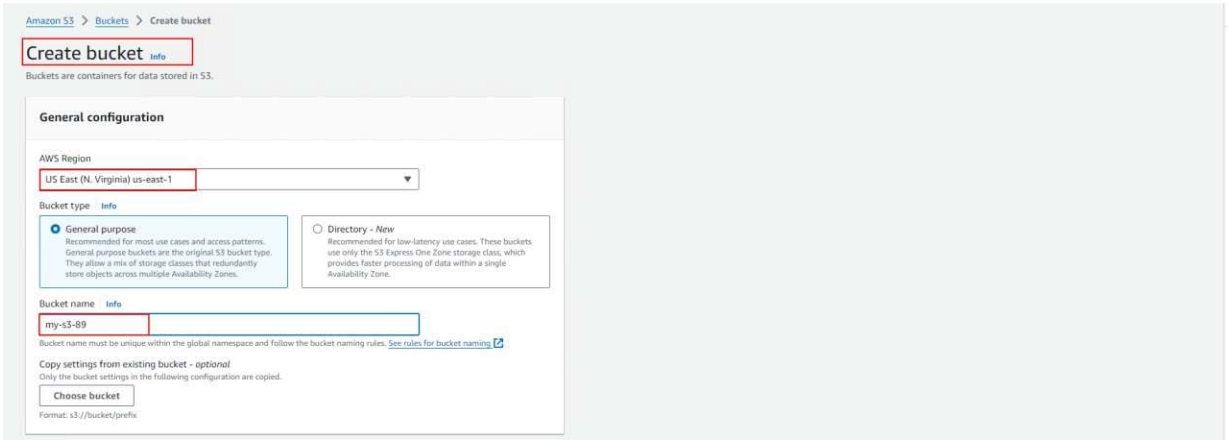
To use S3, type **"S3"** in the search bar and then click on **S3** from the results.



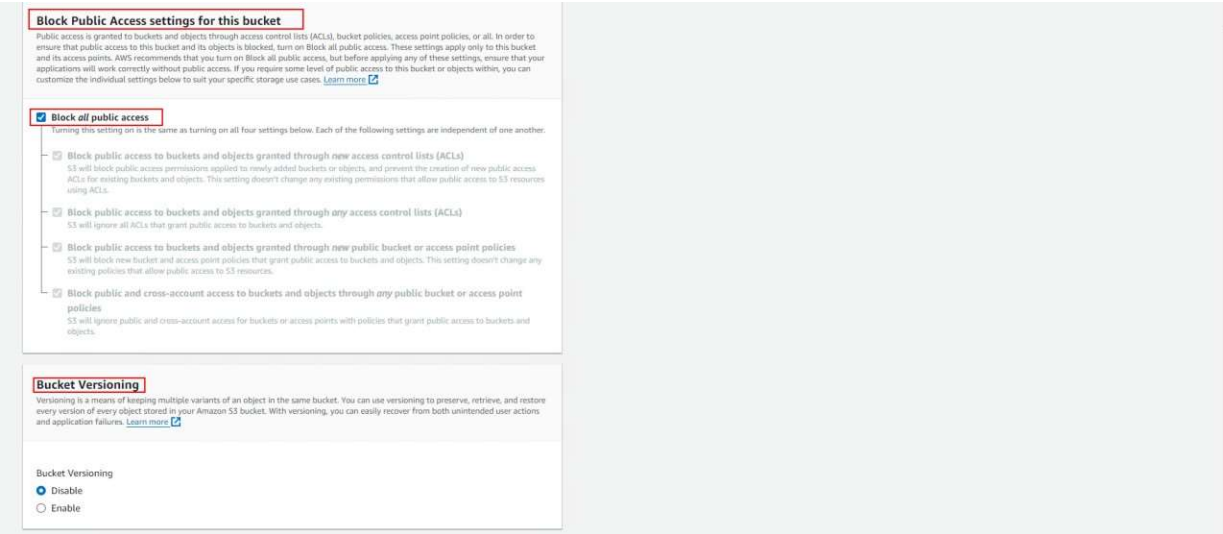
At this stage, click on the **Create Bucket** button.



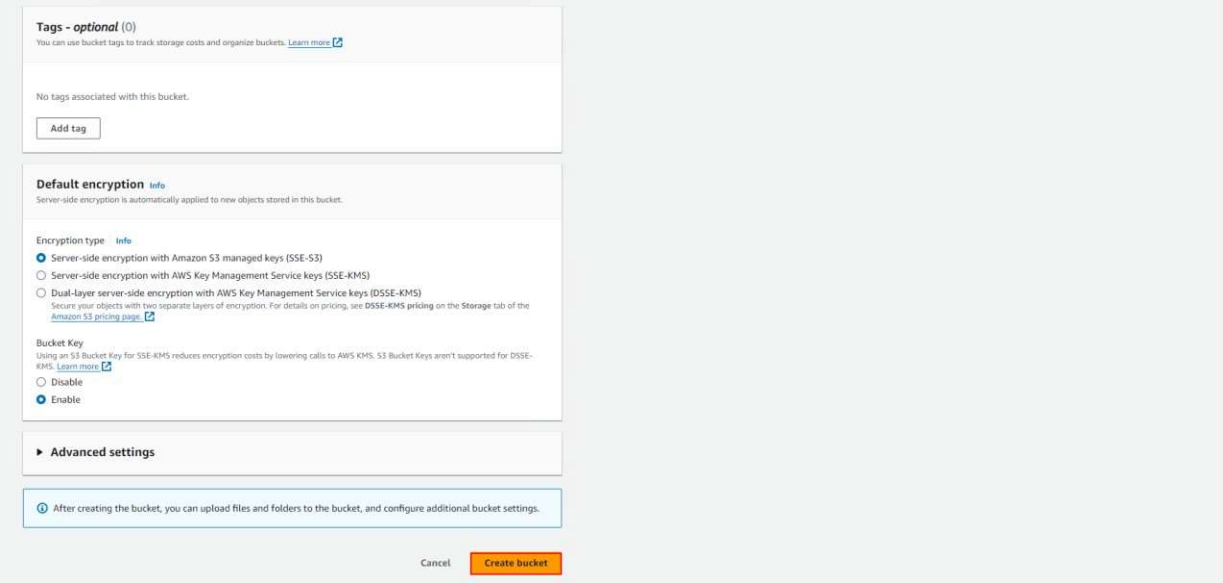
At this stage, you need to specify an **AWS Region** and a **Bucket Name**.



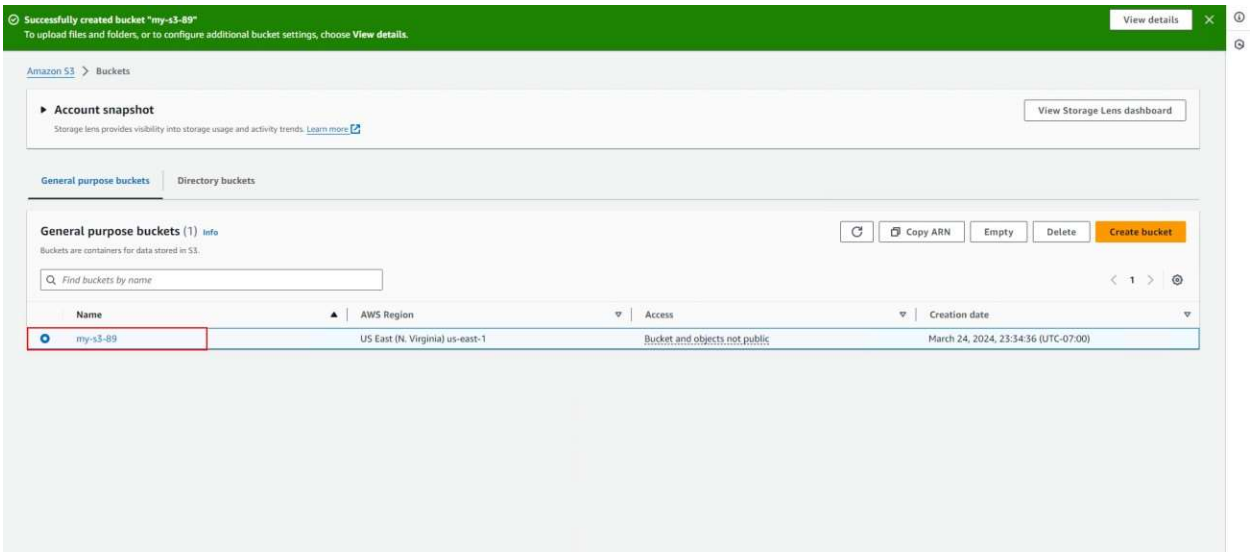
At this stage, if you select the **"Block all public access"** option, access to the S3 bucket from the public network will not be allowed. In this section, you can also enable **Bucket Versioning**.



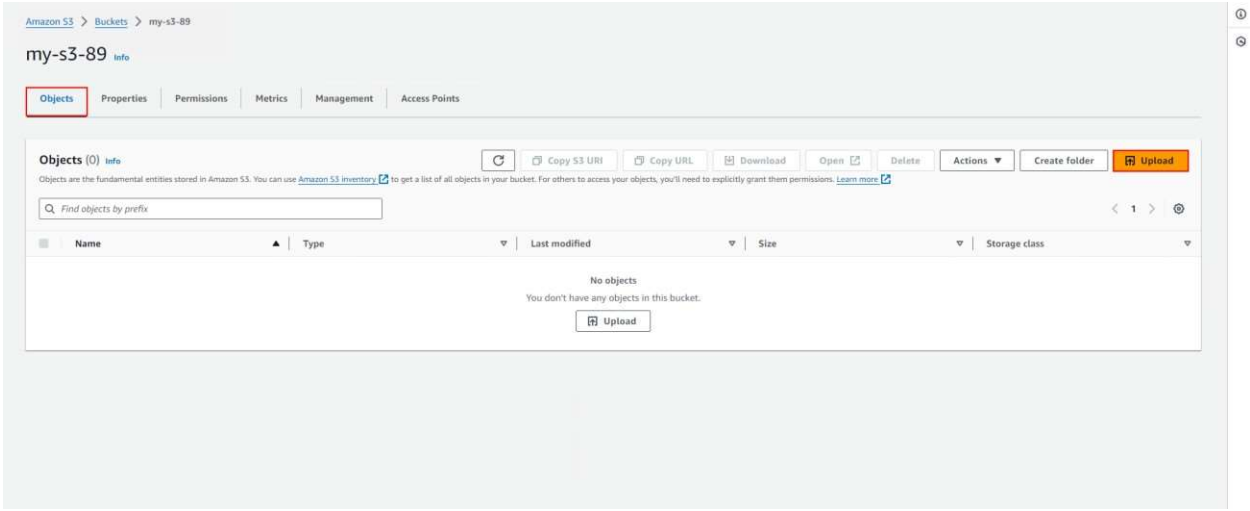
At this stage, click on the **Create Bucket** button.



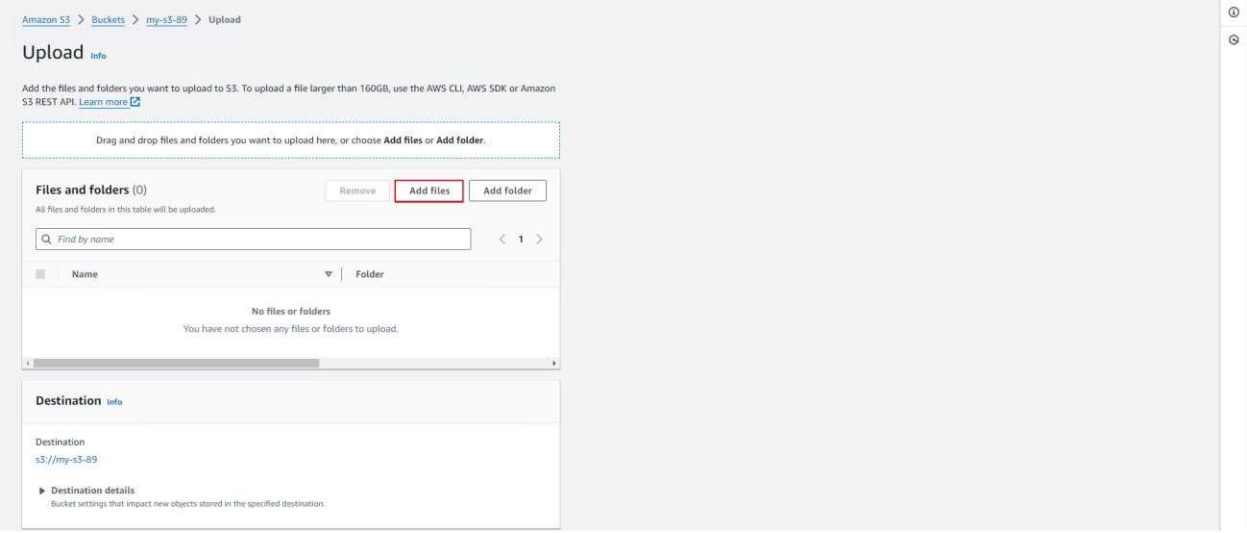
At this stage, select the **bucket** you just created.



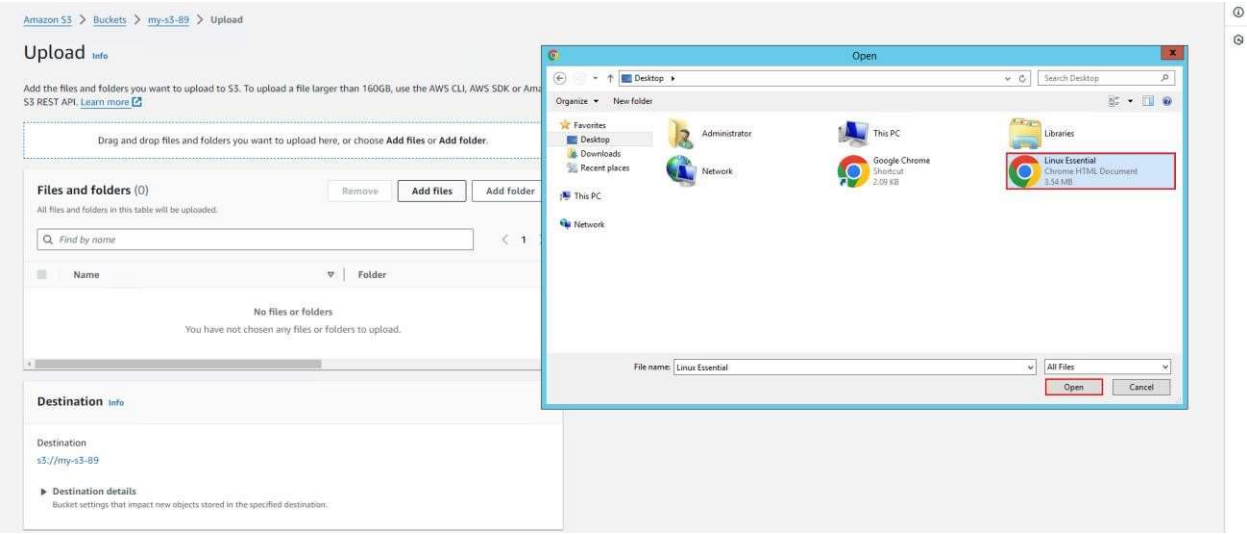
In the **Objects** section, you can upload your files to the **S3 bucket**. To upload a file, click on the **Upload** button.



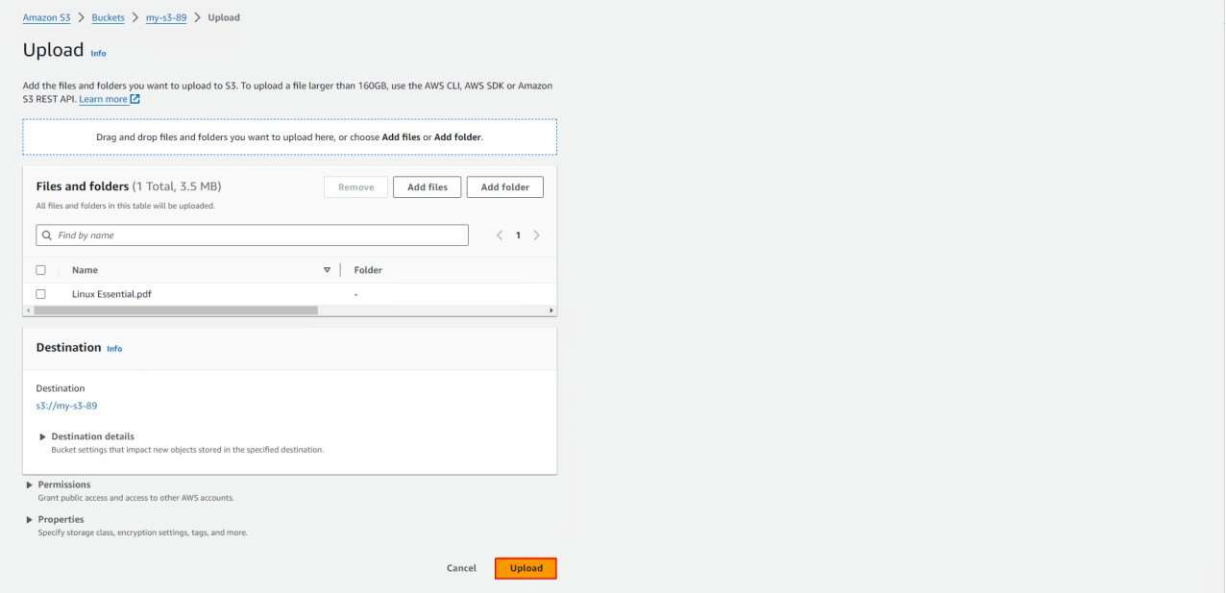
At this stage, click on the **Add Files** button.



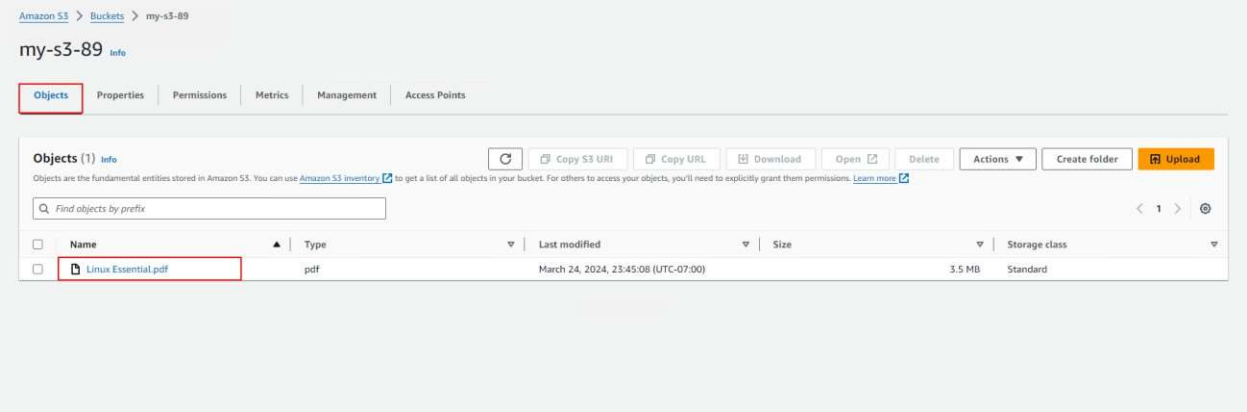
Select your desired file from your computer, then click the **Open** button.



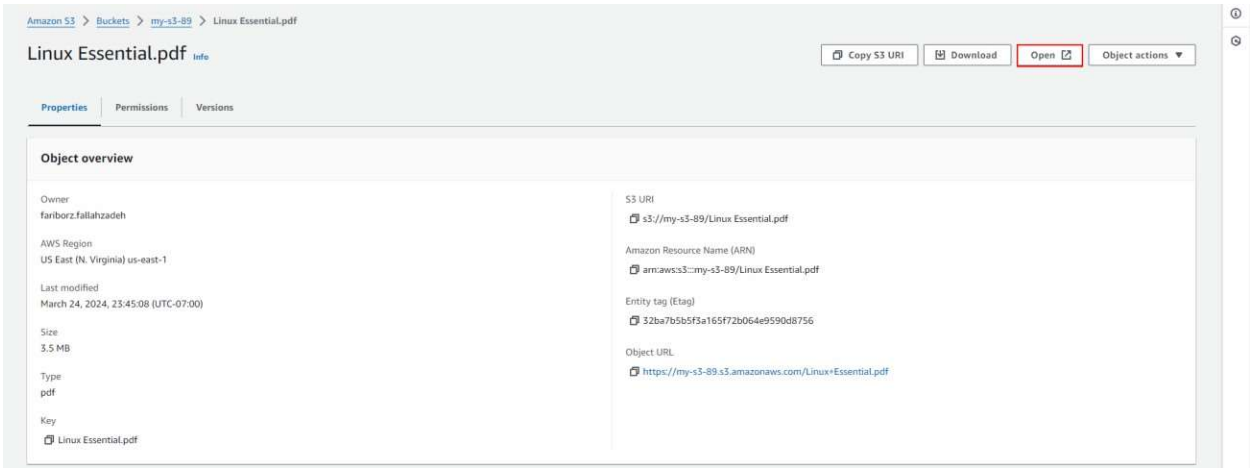
Then, click on the **Upload** button.



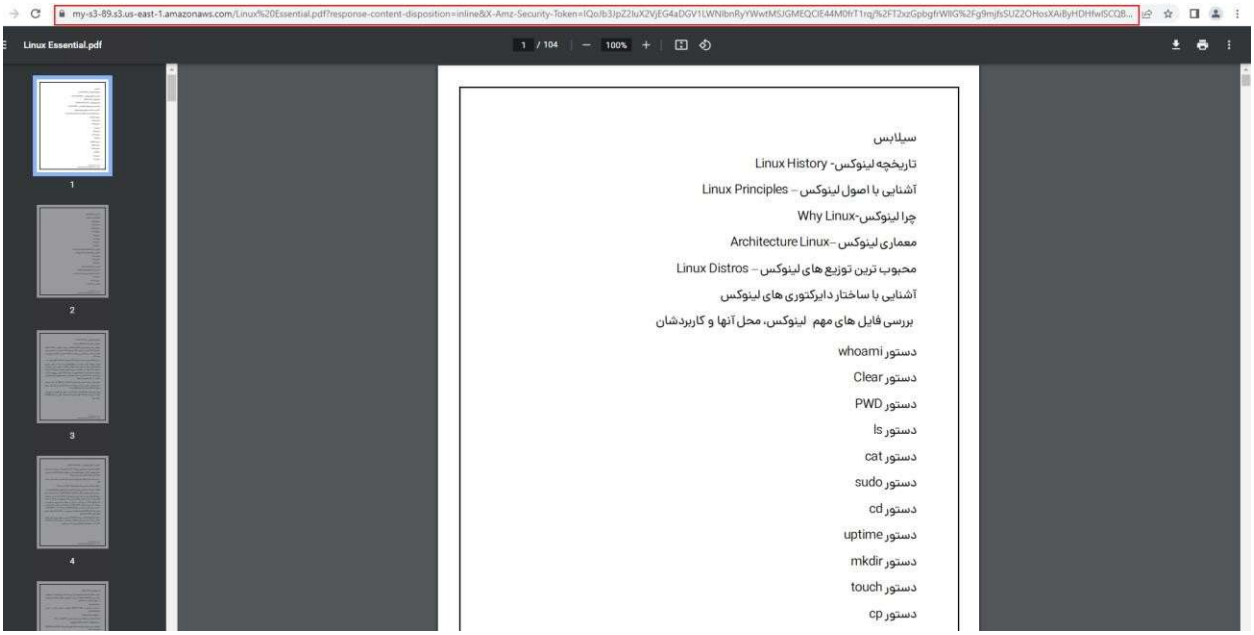
As shown in the image below, our file has been successfully uploaded to the **S3 Bucket**.



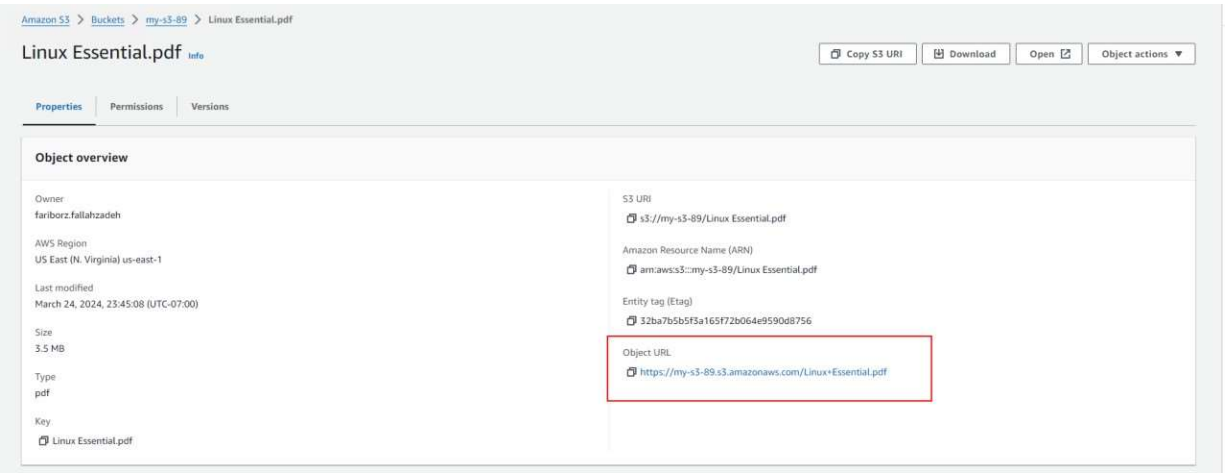
If you click on the **uploaded file**, you will see its details. To open the file, click on the **Open** button.



As you can see, the selected file has been opened on the web.

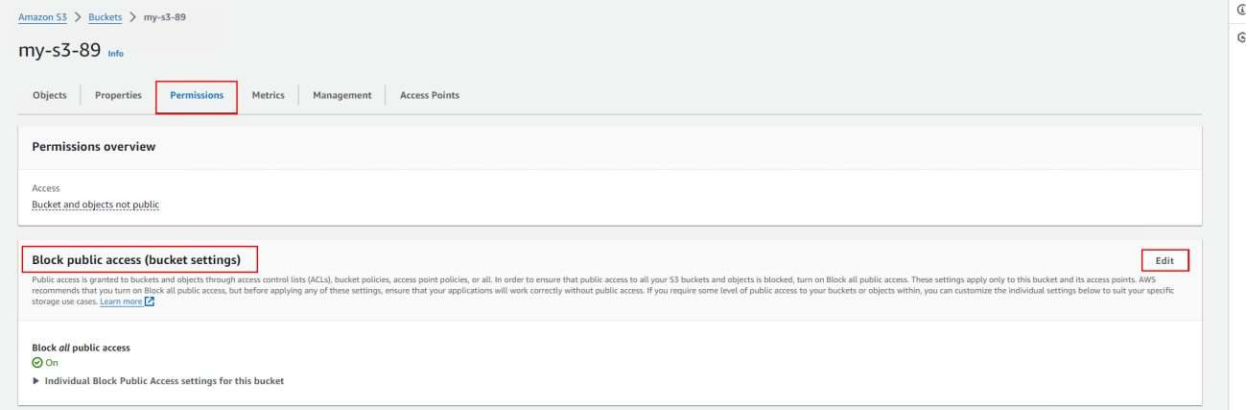


In the file's **Details** section, there is a field called **Object URL**, which is the access link to the file. Copy this URL and paste it into the **address bar** of a browser on your system.

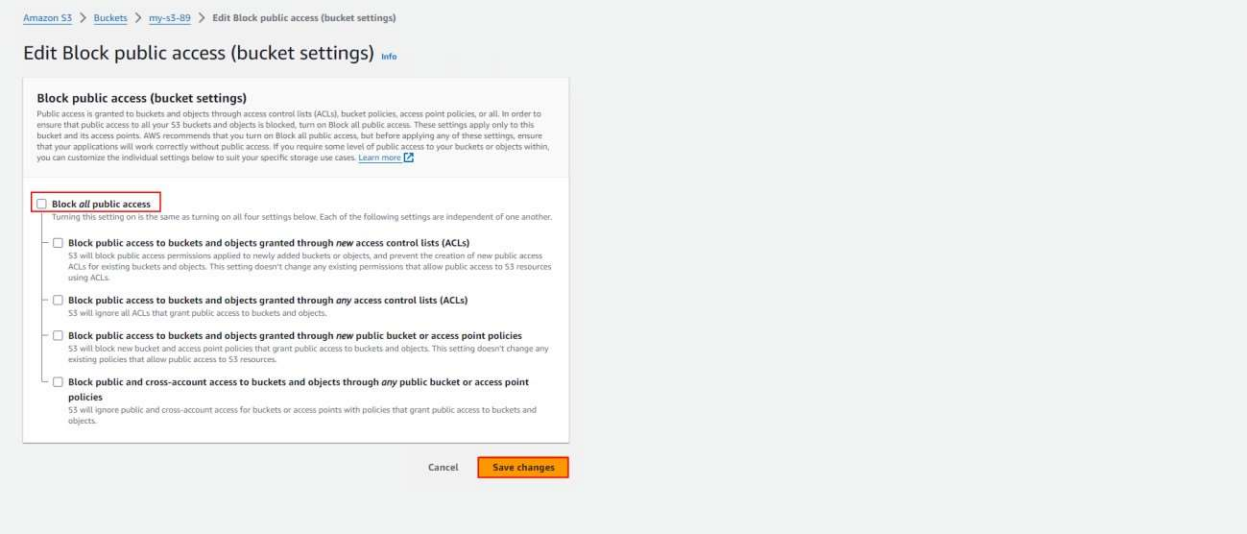


As you can see, access to the file through the **public network** is not allowed.

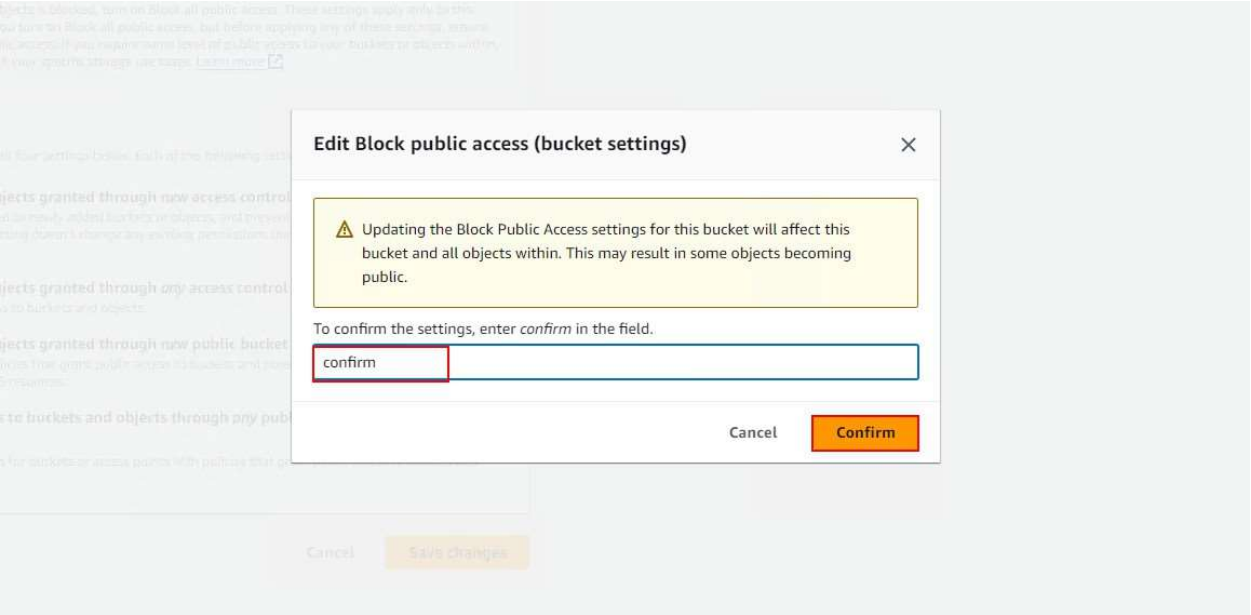
To enable **public access** to the file, go to the **Permissions** tab of the selected file, then click on the **Edit** button under the **Block Public Access** section.



At this stage, **uncheck** the option **Block all public access**, and then click on the **Save Changes** button.

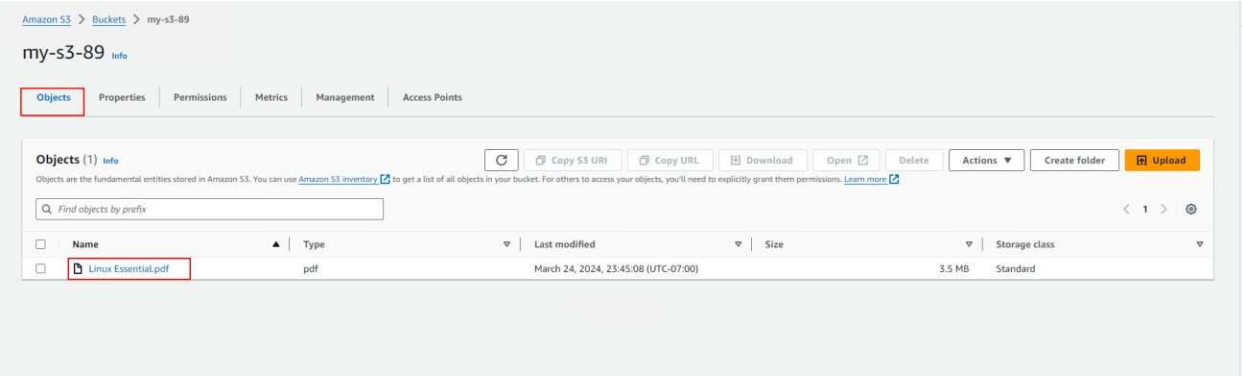


Next, type the word **Confirm** in the provided text box, then click the **Confirm** button.

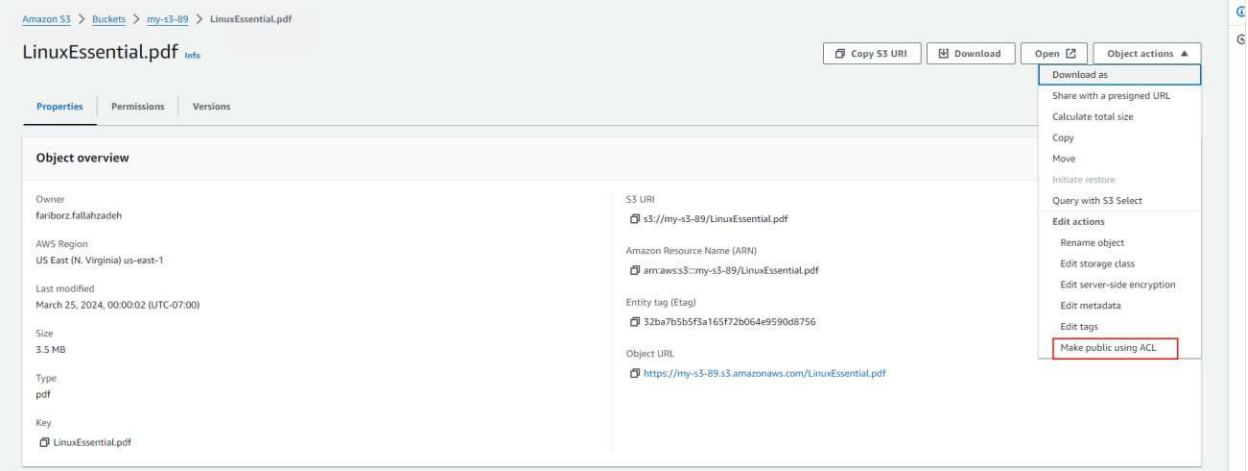


As you can see, when testing web access to the file again, **access is still not allowed**.

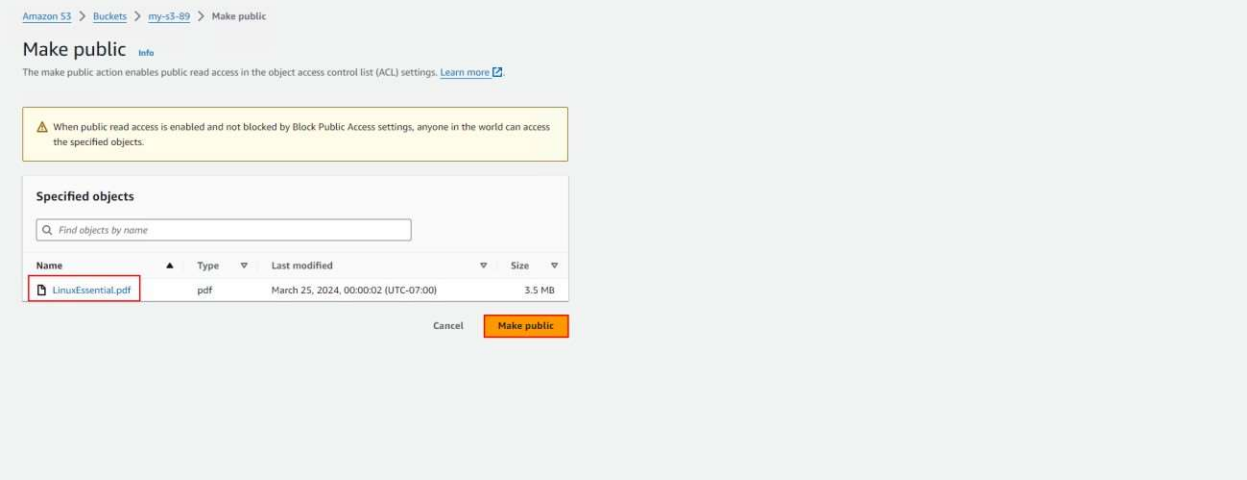
To resolve the file access issue, click on the **desired file**.



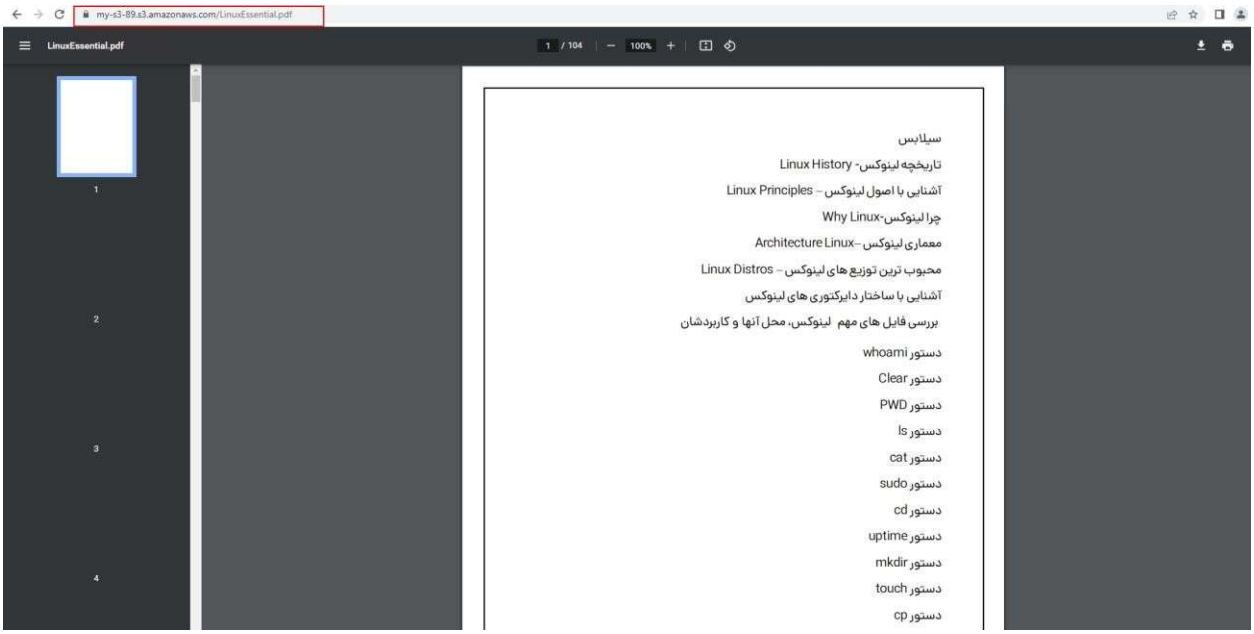
Then, in the **Object actions** menu, click on **Make public using ACL**.



Then, click on the **Make Public** button.



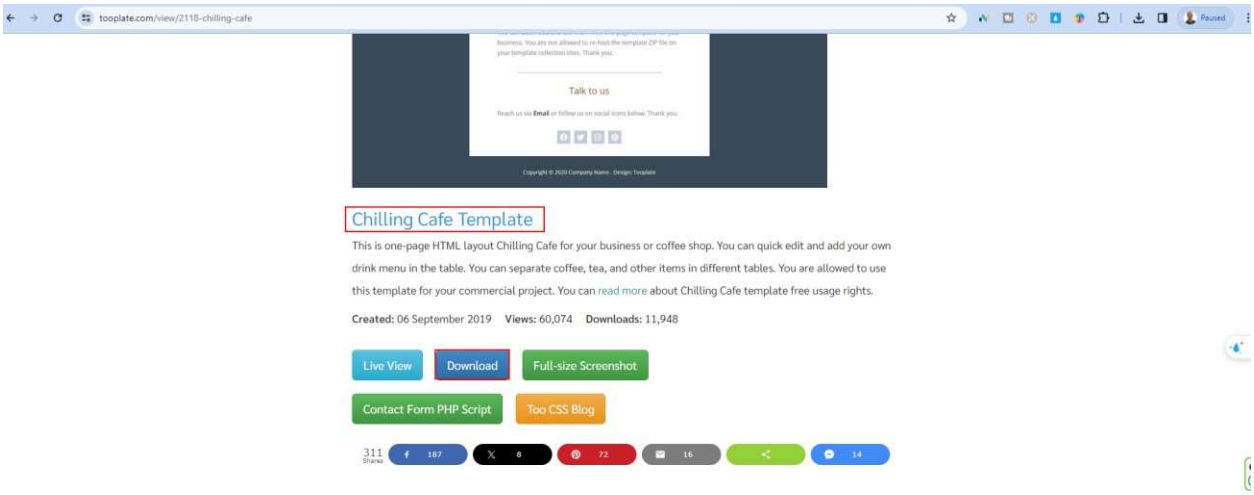
As you can see, the file is now **accessible via the web** and through the **public network**.



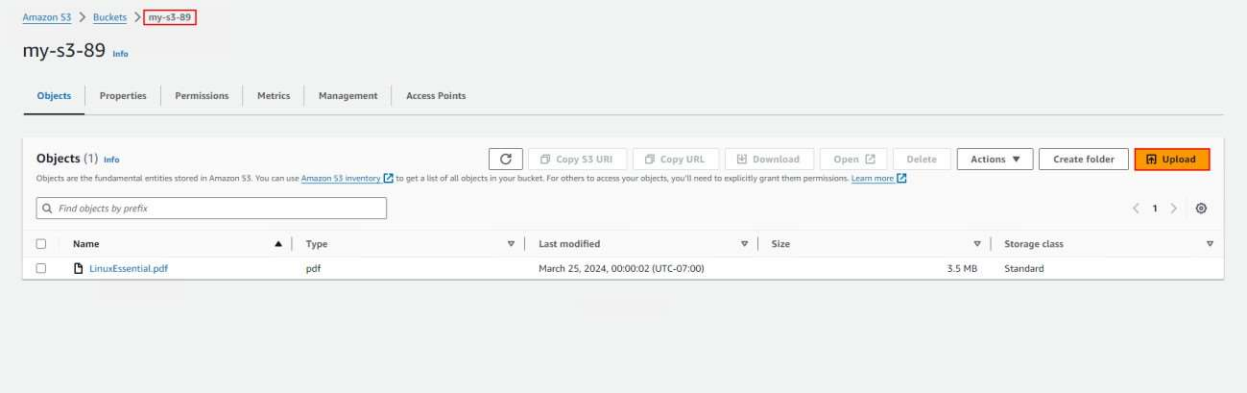
How to Host a Static Website Using an S3 Bucket

We can use an **S3 Bucket as a host** for web files as well.
To better understand this, we'll use a **template from the Tooplate website**, download one of the templates, upload it to an **S3 bucket**, and use that **S3 bucket as a web server**.

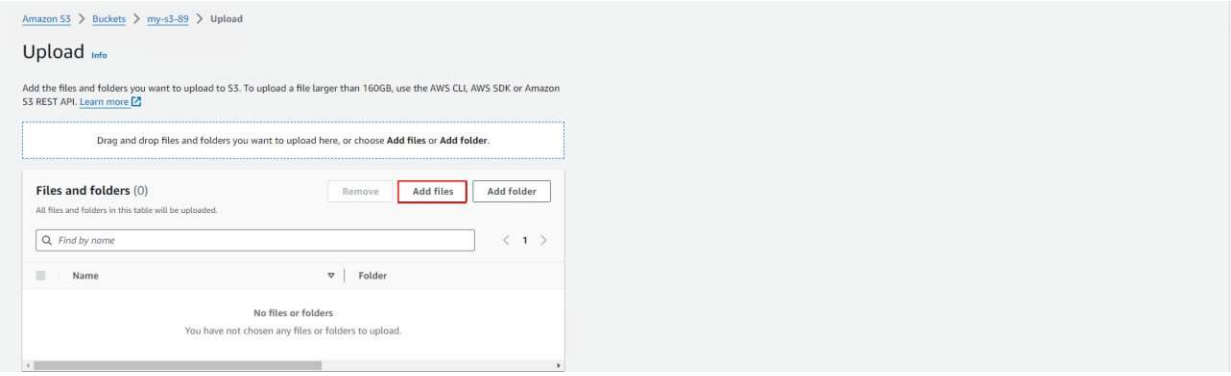
First, go to the **Tooplate website** and download one of the **template files**.



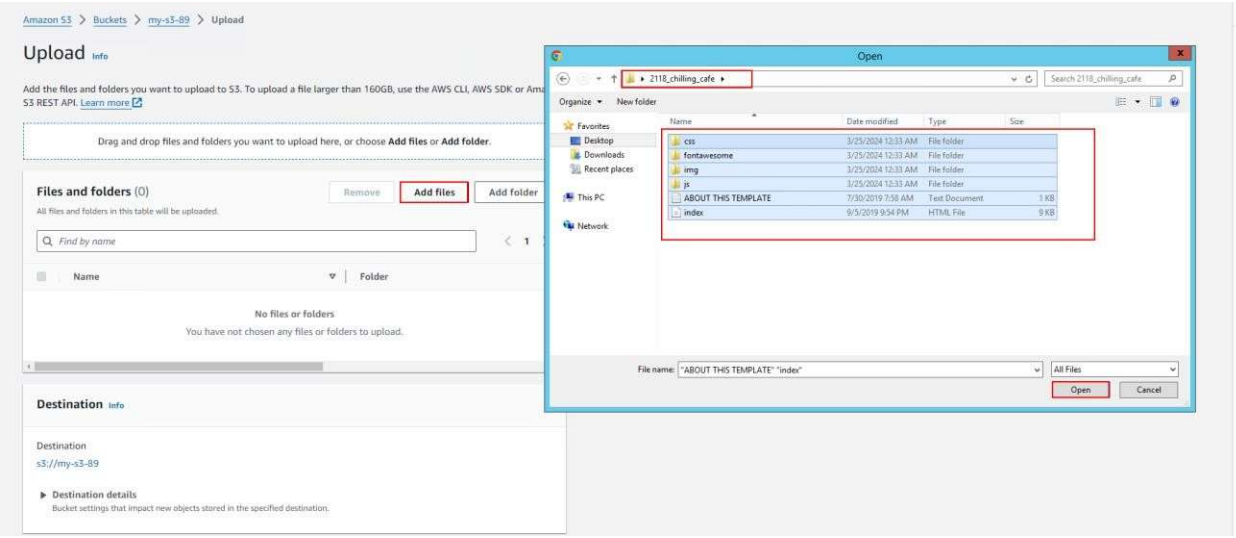
Then, go to the **S3 bucket** you created in the previous steps and click on the **Upload** button.



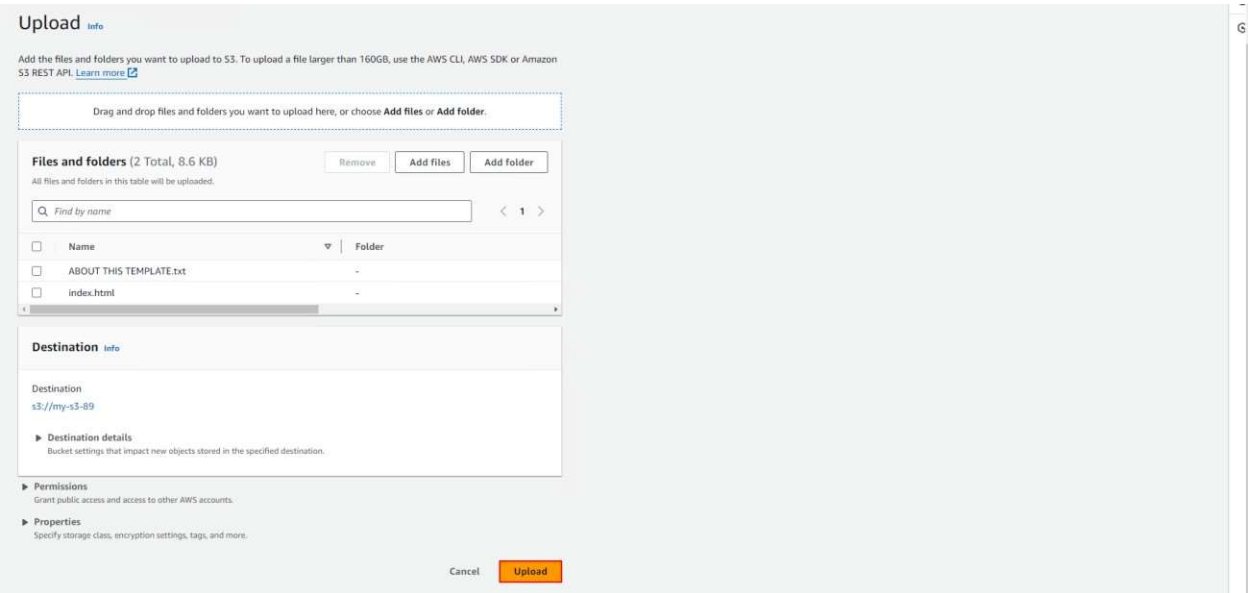
At this stage, click on the **Add Files** button.



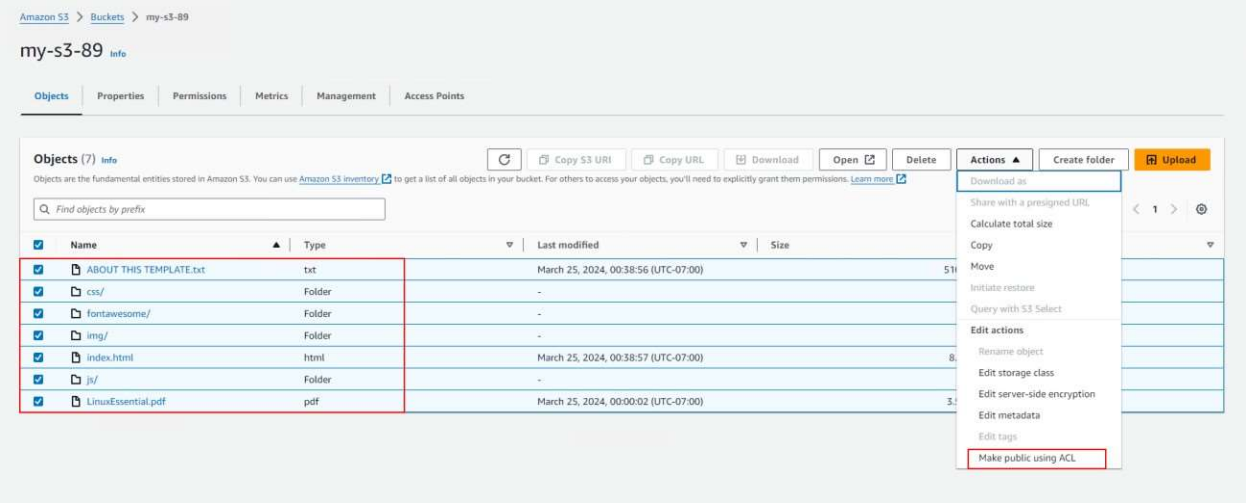
Then, select the files and click the **Open** button.



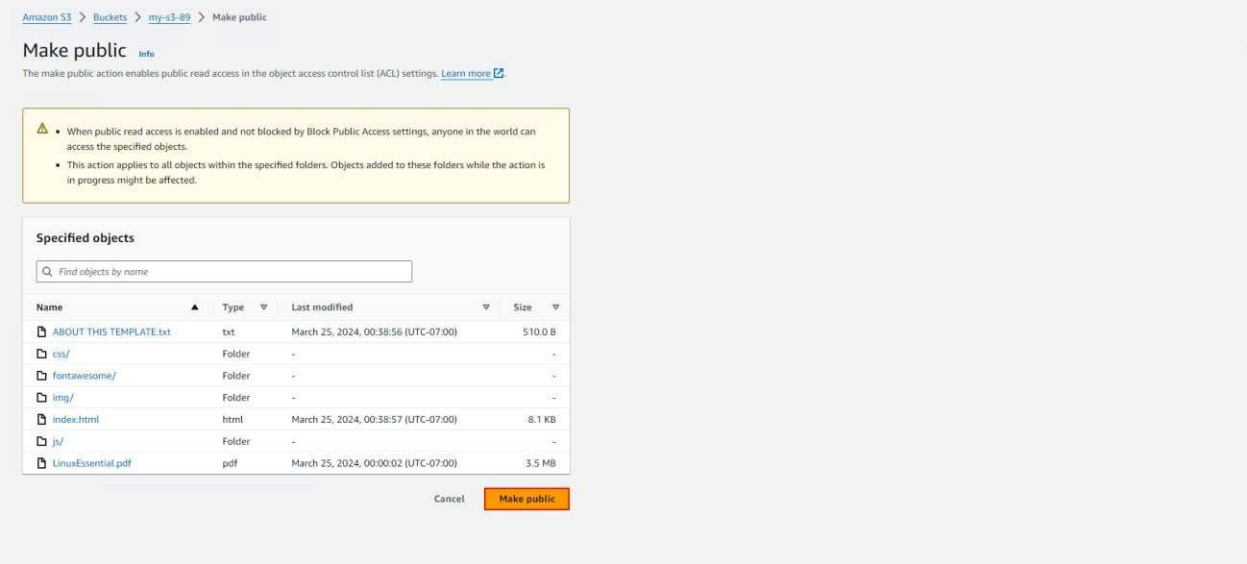
Then, click on the **Upload** button.



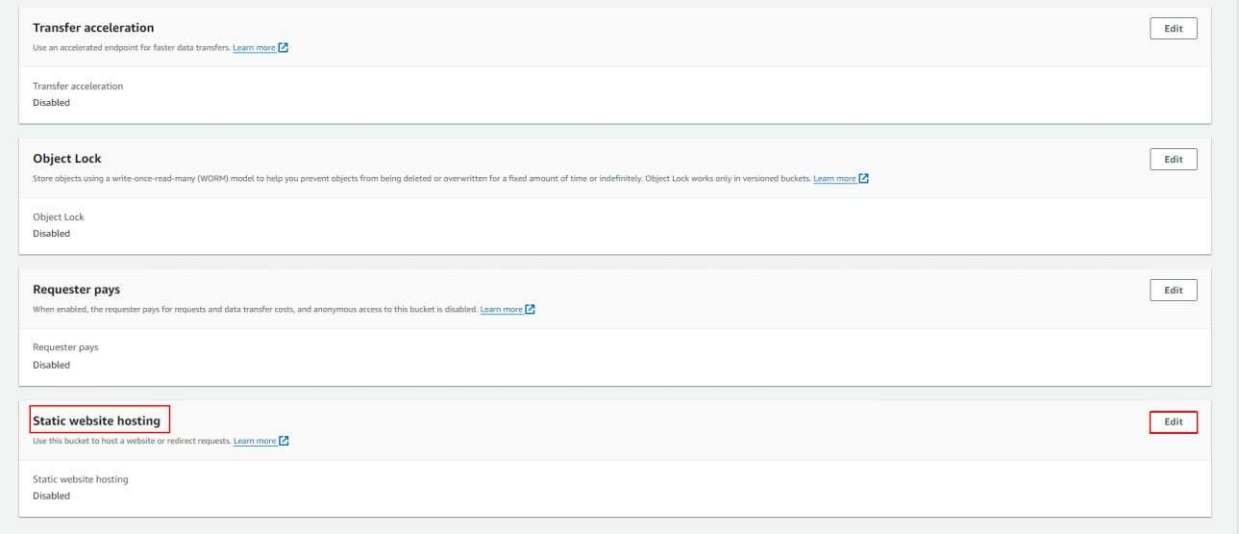
At this stage, select all the **template files**, then from the **Actions** menu, choose **Make Public Using ACL**.



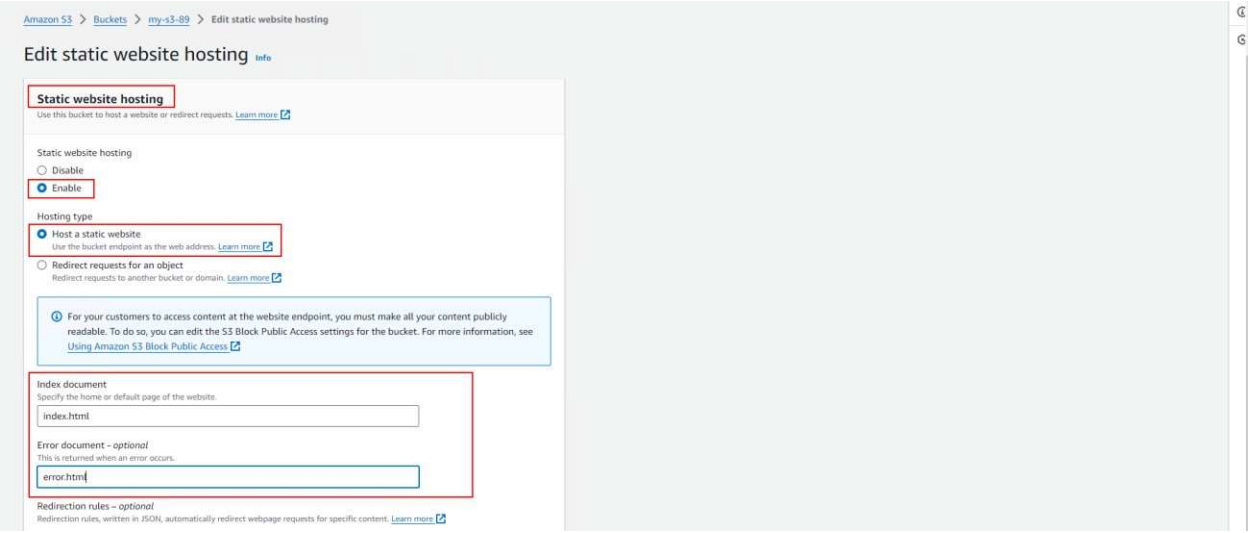
Next, click on the **Make Public** button.



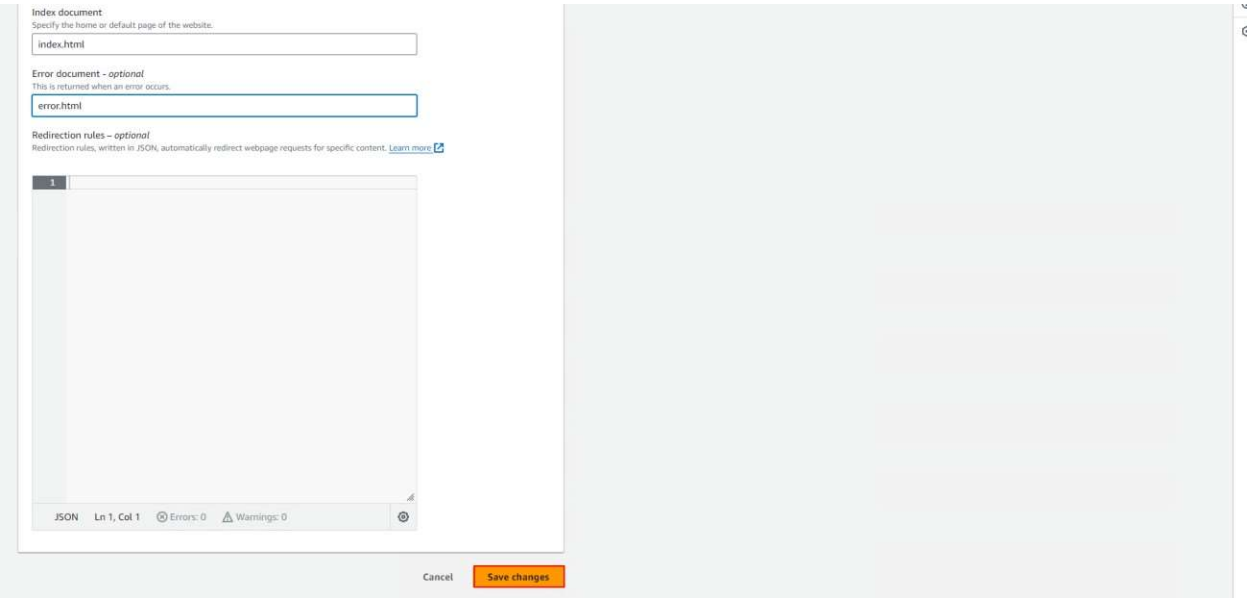
Then, in the **S3 Bucket settings**, go to the **Static Website Hosting** section and click the **Edit** button.



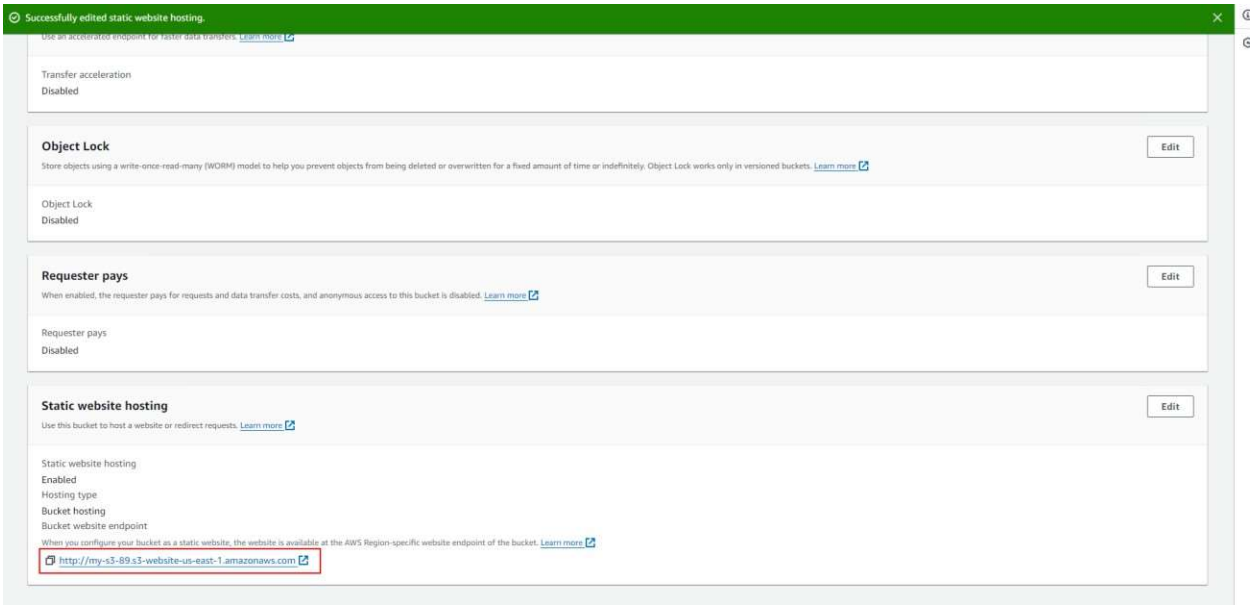
Then, enter the following settings in this section



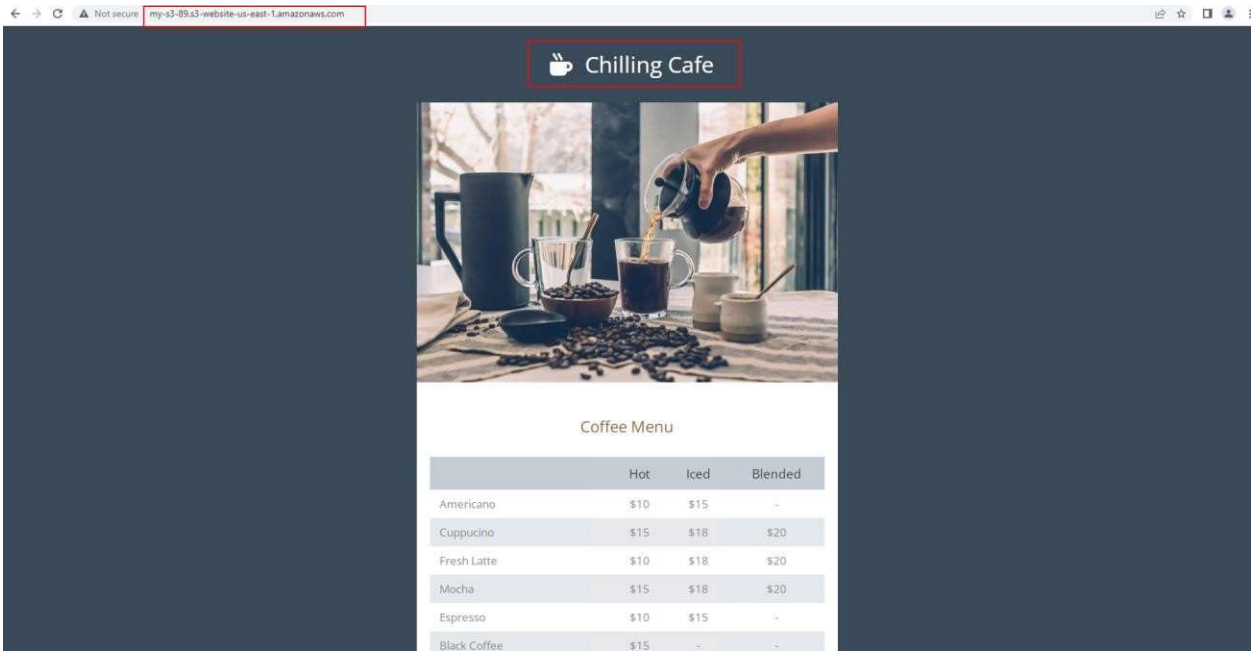
And finally, click on the **Save Changes** button.



As you can see, after completing the settings, a **URL** is provided which allows access to your **website files**.



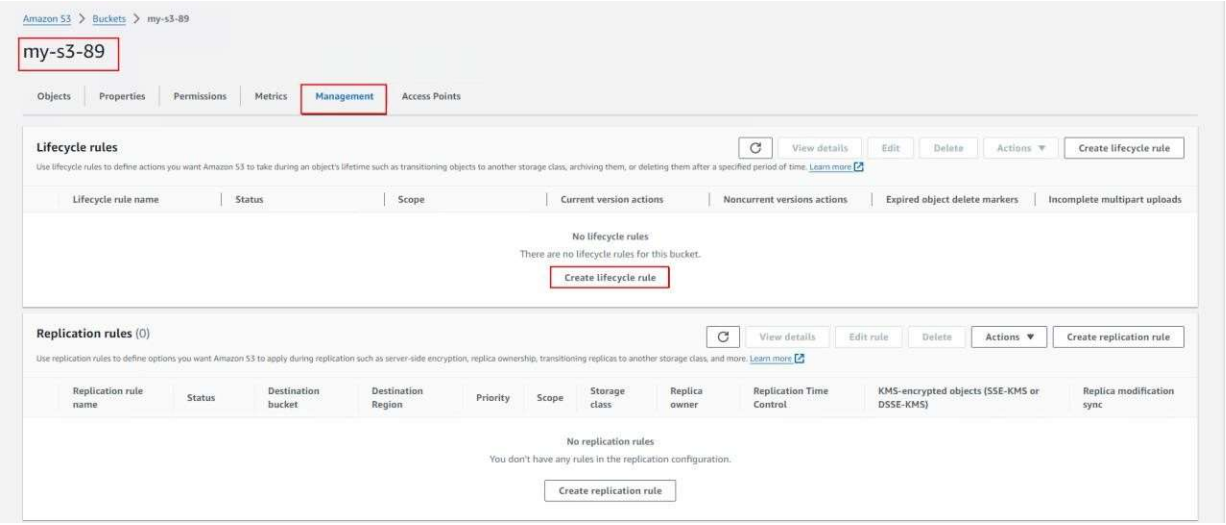
When you enter the above link in your **browser**, you will see the **desired website** that is now **hosted on your S3 bucket**.



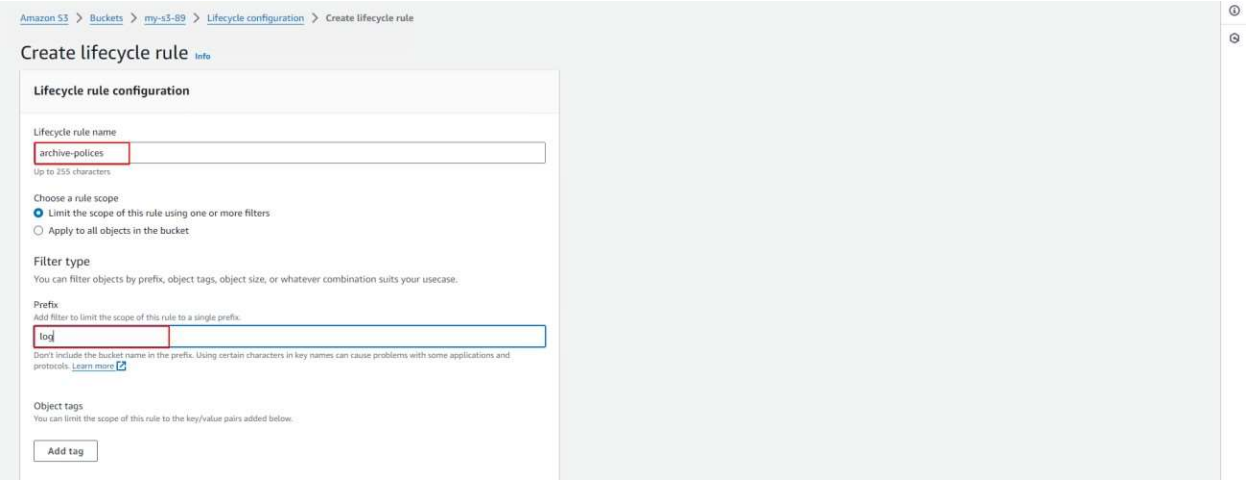
How to Use S3 Bucket Lifecycle

You can define a **Lifecycle** for an **S3 Bucket**, which means your files will transition from one **storage class** to another over time and eventually be **deleted**.
Using a **lifecycle** helps **reduce costs** for maintaining files within the **S3 bucket**.

To define an **S3 Bucket Lifecycle**, click on your **S3 bucket**, then go to the **Management** tab and click on the **Create lifecycle rule** button.



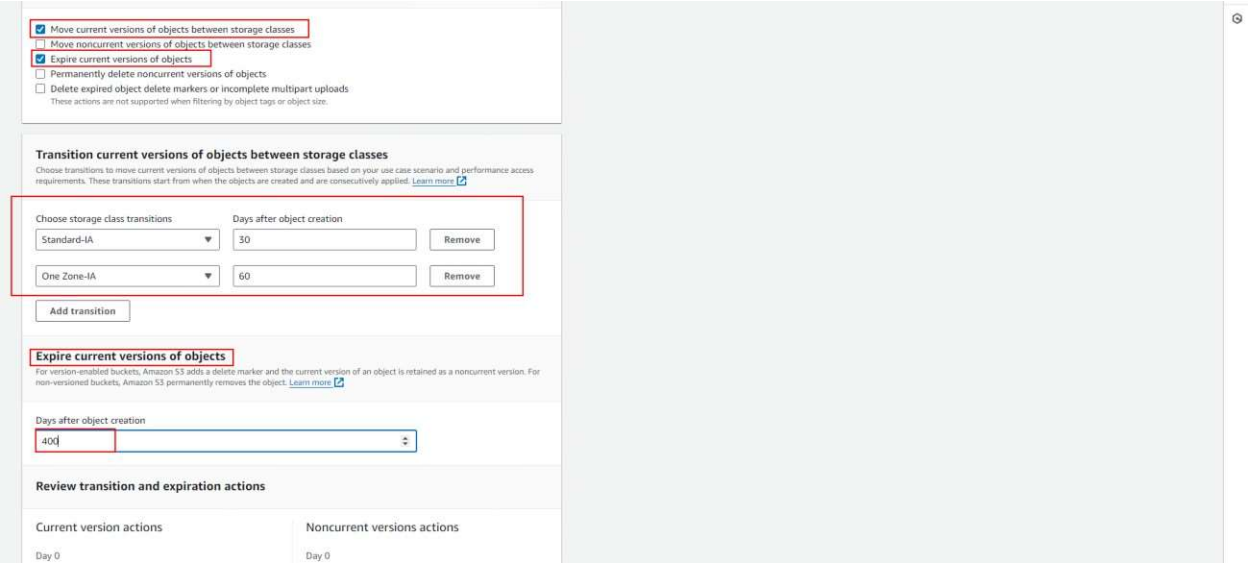
At this stage, specify a **name** for the lifecycle rule and define a **prefix**.



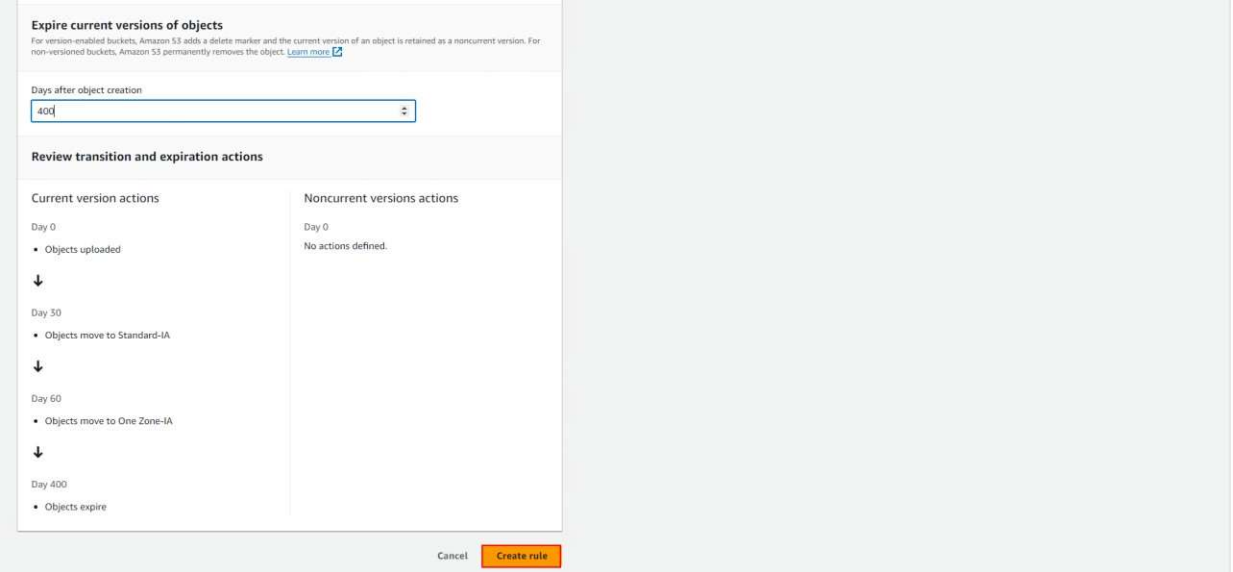
At this stage, select the option **"Move current version of objects between storage classes"** to enable automatic transitions between storage classes.
Then, select **"Expire current version of objects"** to define after how many days the data should be **deleted**.

Then, you need to specify the **number of days** the data should remain in each **storage class** before being transitioned or deleted.

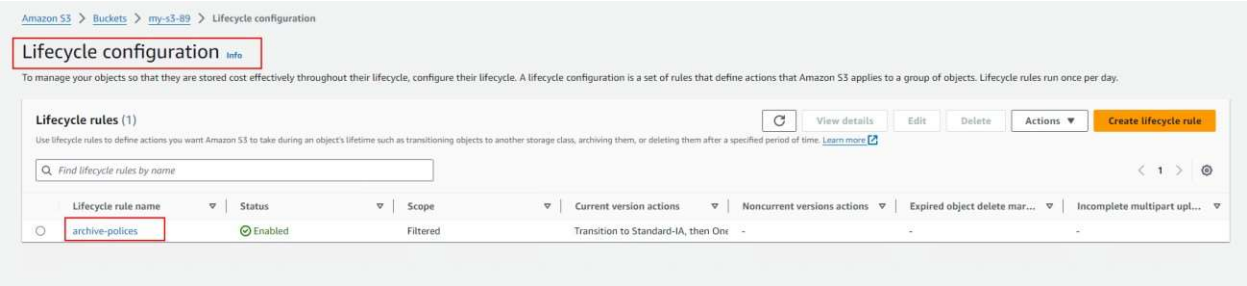
In this step, you must define how many **days** the data should remain in each **storage class** before it should **transition** to the next storage class.



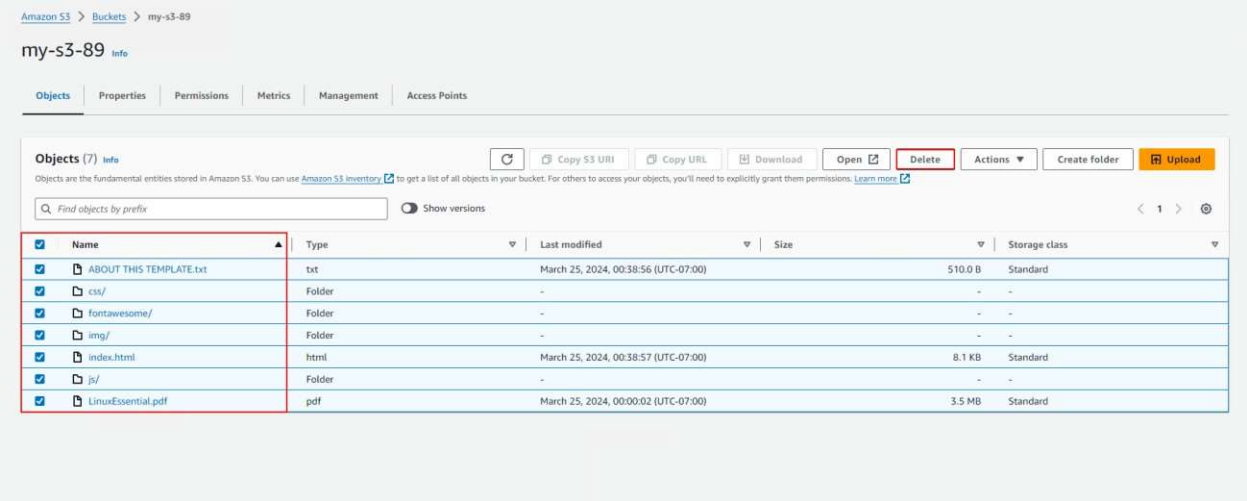
And finally, click on the **Create Rule** button.



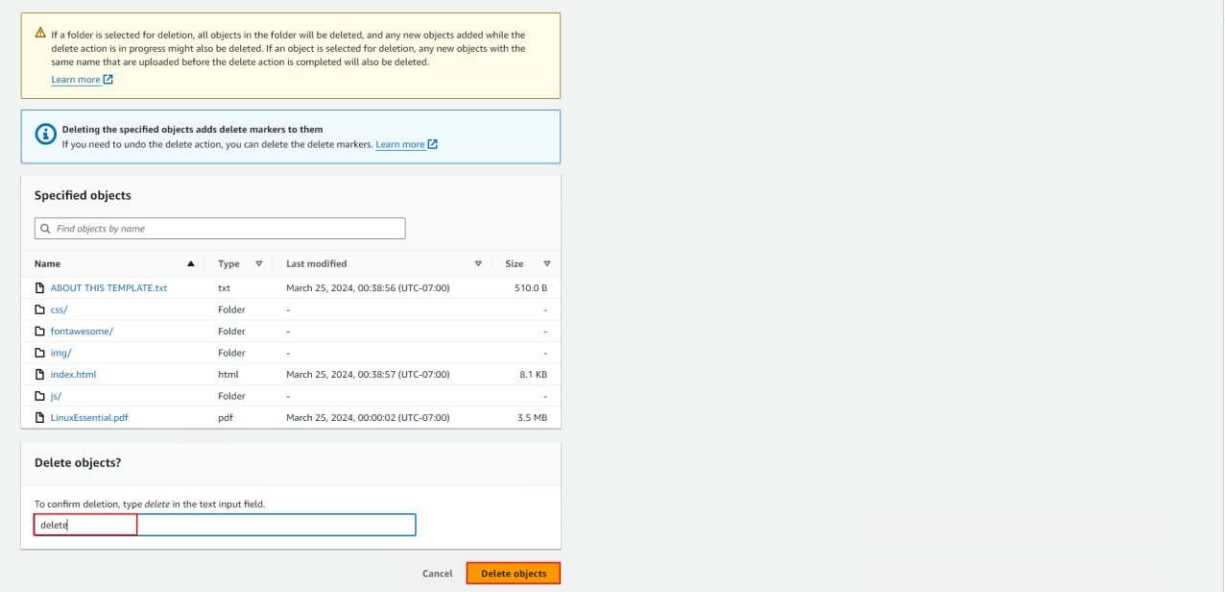
As you can see, the **lifecycle rule** has been successfully created for our **S3 bucket**.



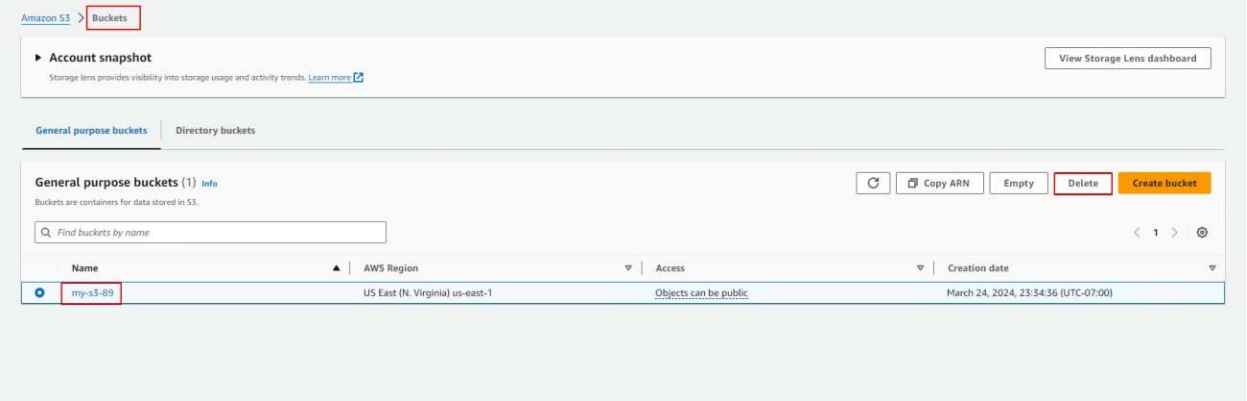
To delete files from an **S3 bucket**, select the files and then click the **Delete** button.



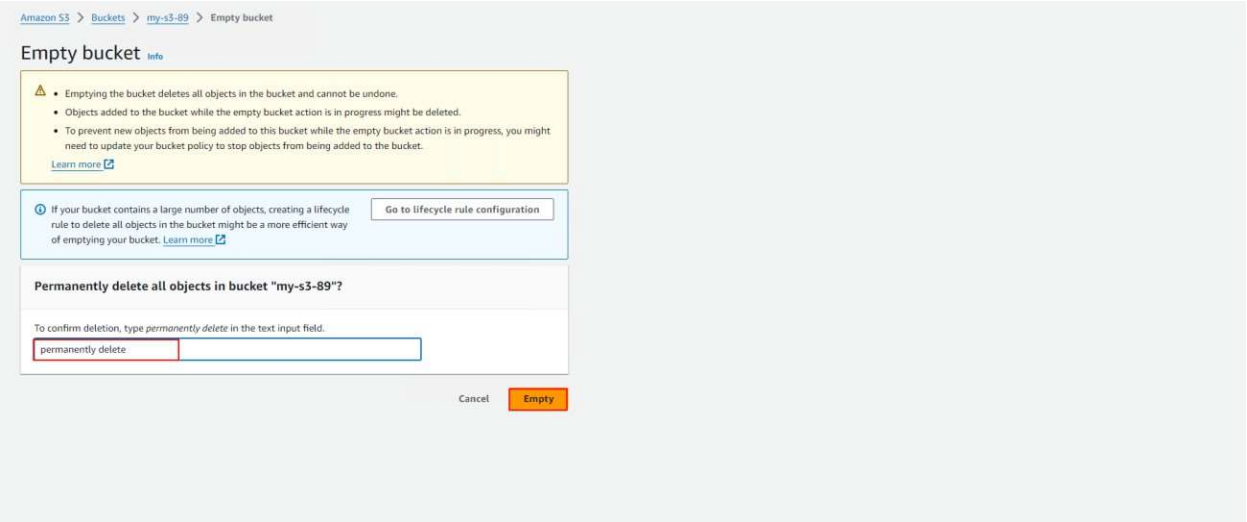
In the text box, type the word **Delete**, then click on **Delete objects**.



To delete an **S3 bucket**, select the bucket and then click on the **Delete** button.



In the text box, type **Permanently Delete**, then click the **Empty** button.



Introduction to AWS RDS

AWS RDS is a **managed database service** provided by Amazon that makes it easy to set up, operate, and scale a **relational database** in the cloud.

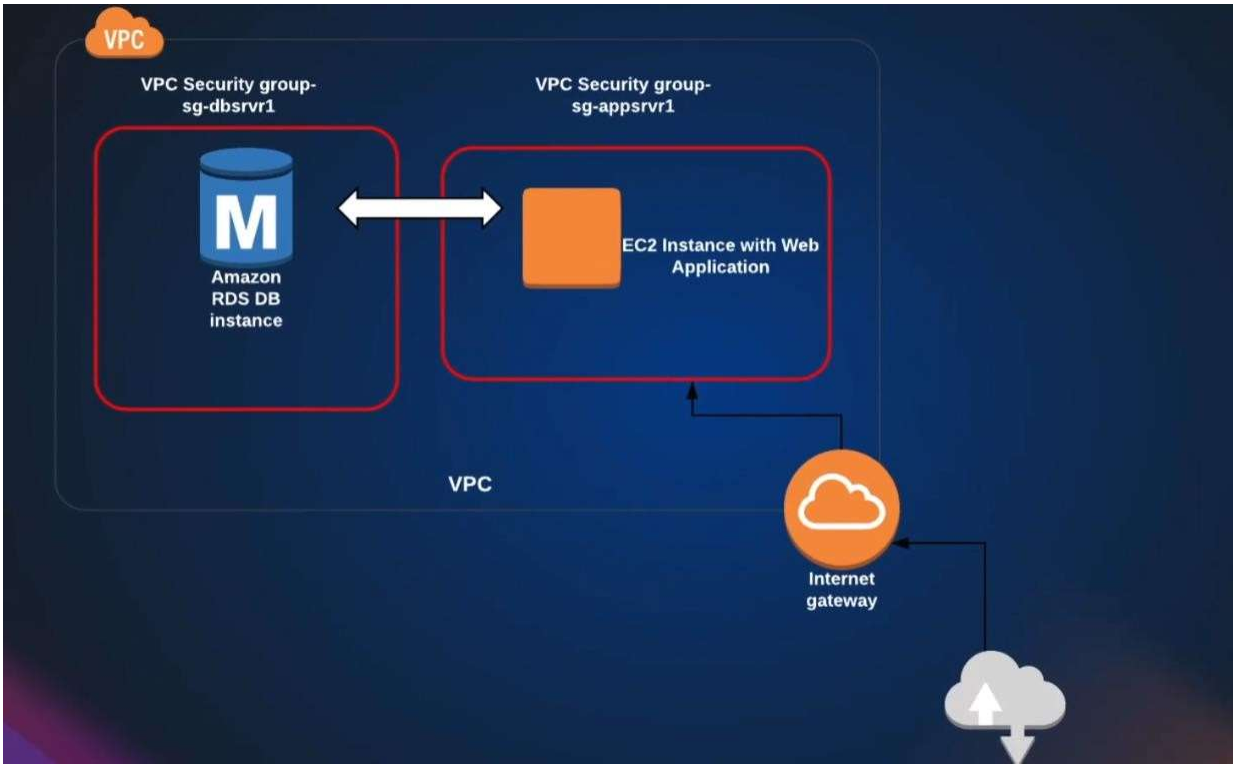
Key Features of AWS RDS:

- **Fully managed**: AWS handles backups, patching, monitoring, and scaling.
- **Supports multiple database engines**, including:
    - Amazon Aurora
    - MySQL
    - PostgreSQL
    - MariaDB
    - Oracle
    - SQL Server
- **High availability** with Multi-AZ deployment.
- **Automated backups and snapshots**.
- **Scalable performance**: You can easily change compute and storage resources.
- **Security**: Integration with IAM, encryption at rest and in transit, VPC support.

Common Use Cases:

- Hosting **production databases** for web or mobile apps.
- Running **analytical workloads** on structured data.
- Creating **high-availability** database environments with failover.

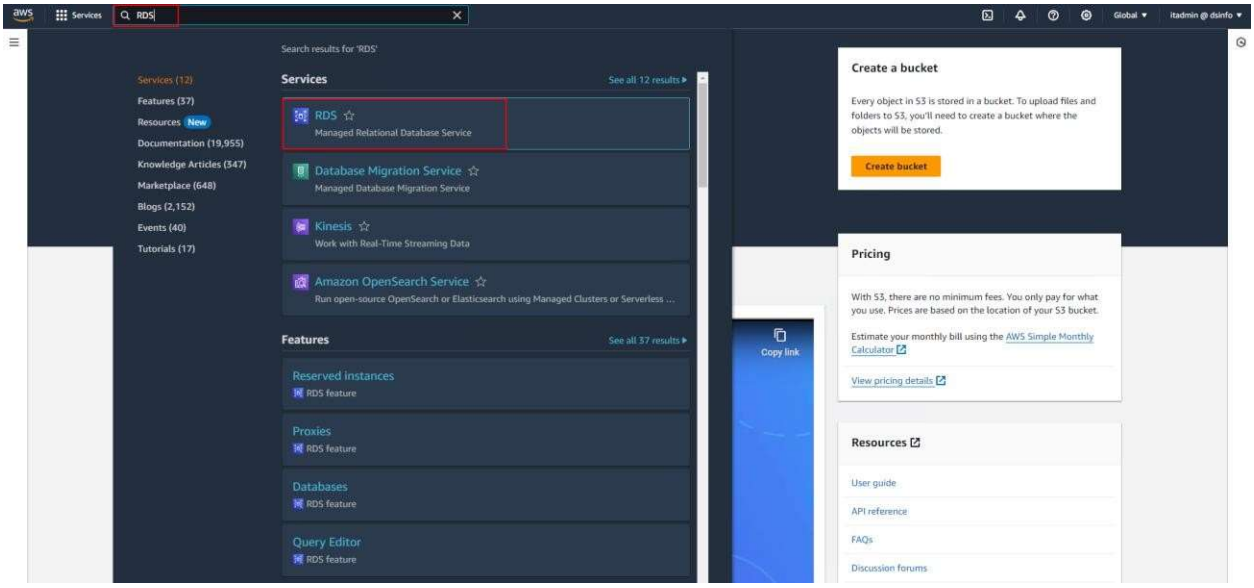
With AWS RDS, you don’t need to worry about the complexity of managing database software or infrastructure — AWS takes care of it for you.



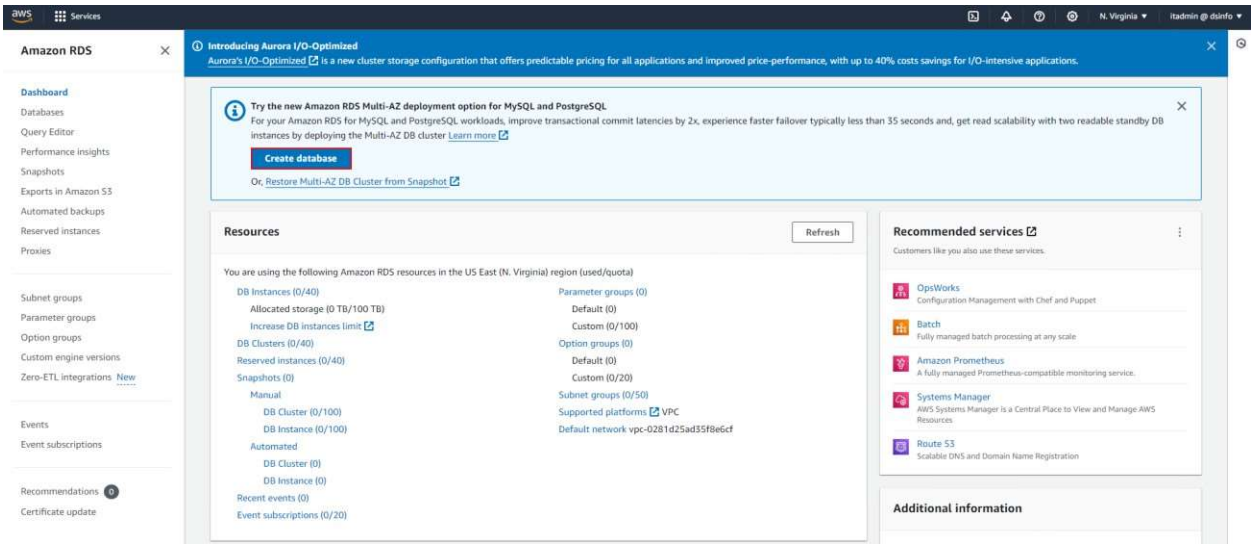
How to Set Up AWS RDS

The **RDS service** supports **various types of databases**.

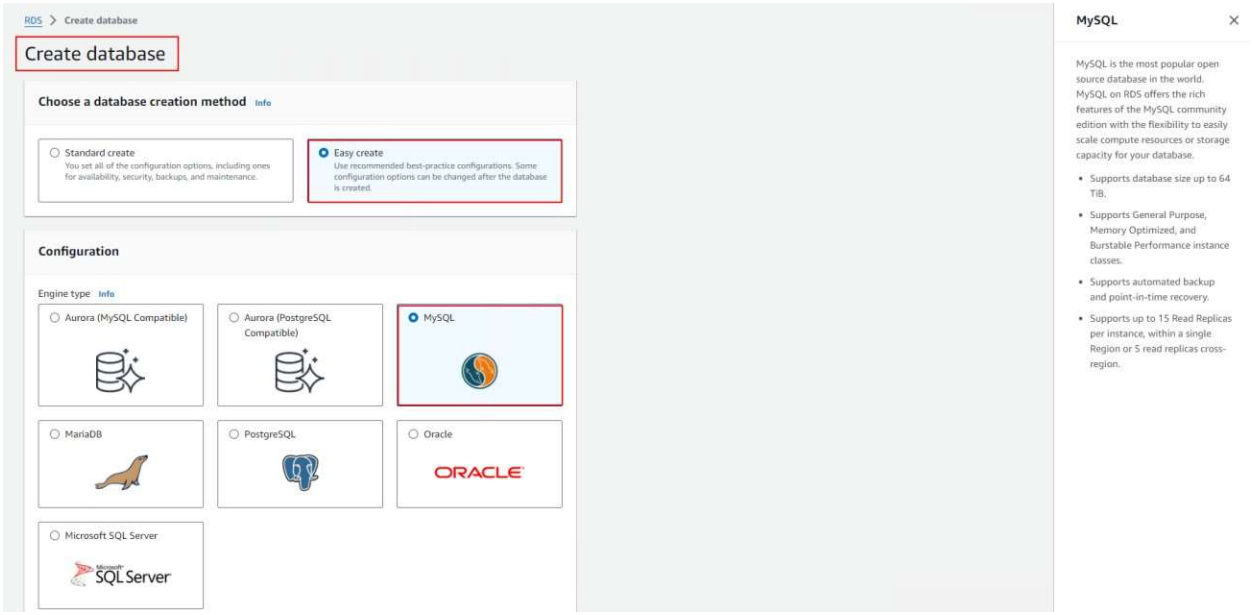
To use **RDS**, type **RDS** in the search bar and then click on **RDS** from the results.



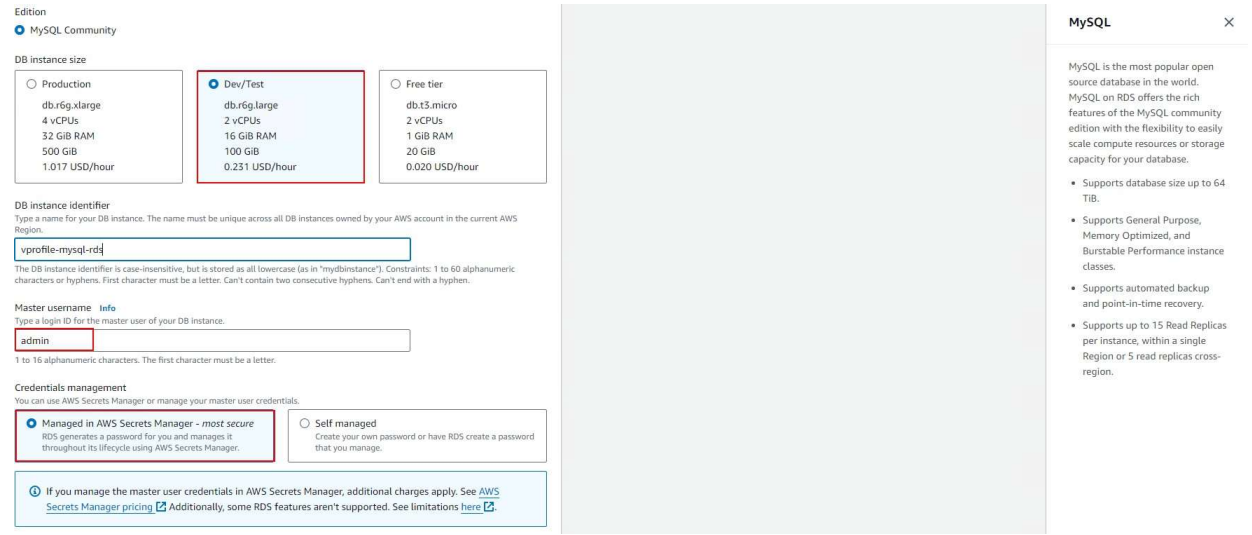
To create a database using **RDS**, click on the **Create Database** button.



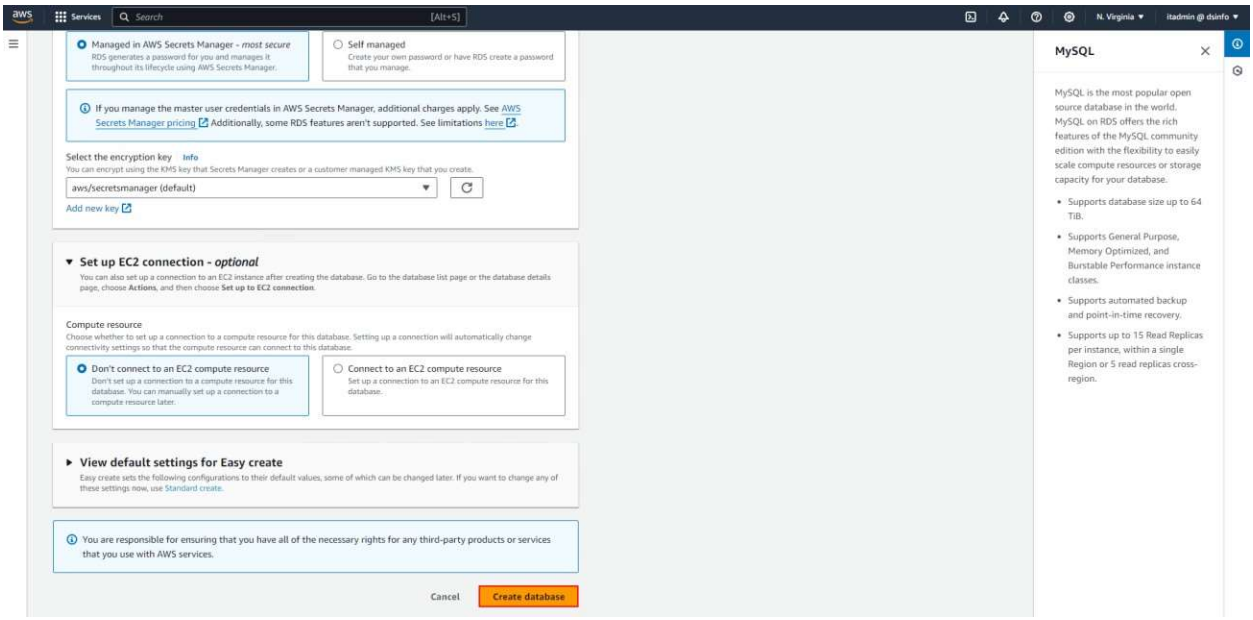
At this stage, you can select your desired **database engine**.



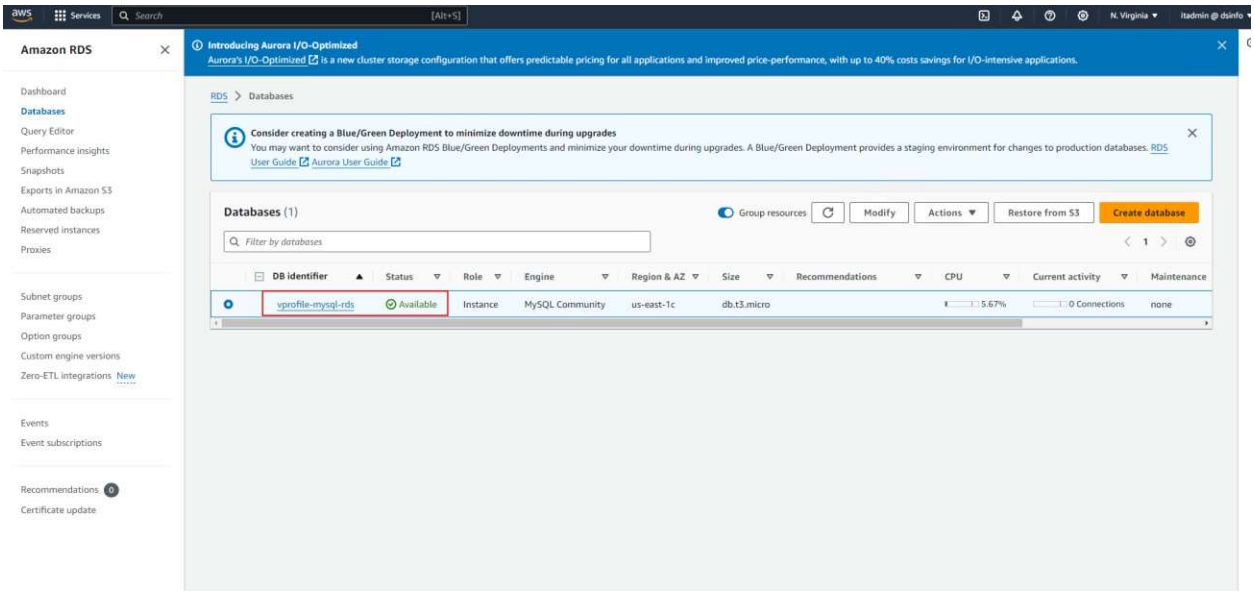
At this stage, you need to specify the required **resources** for the database, along with the **username** and **password** for database access.



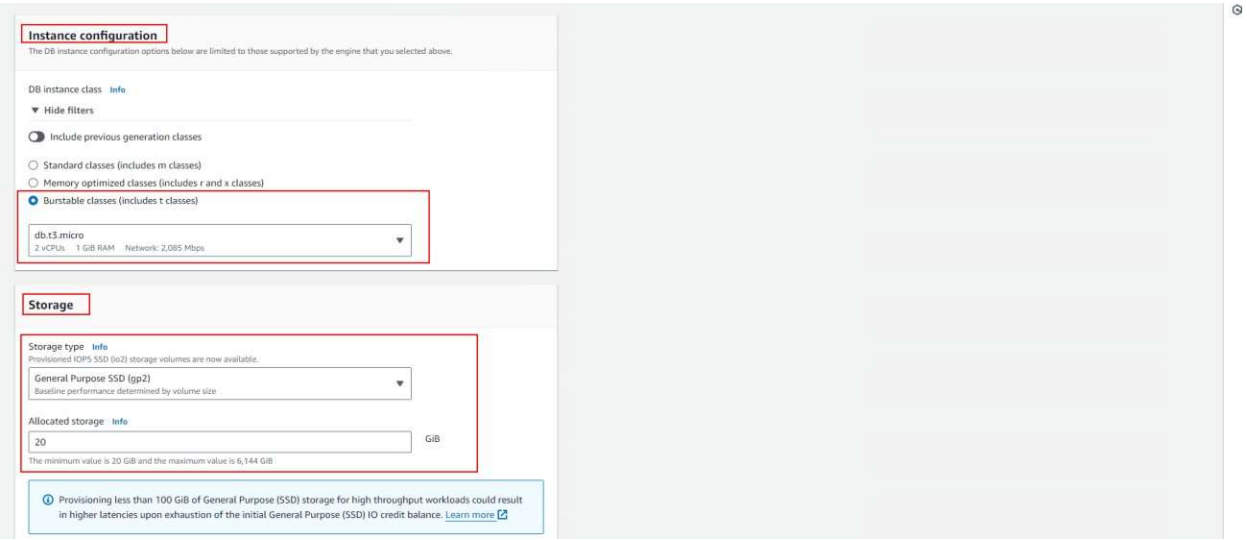
And finally, click on the **Create Database** button.



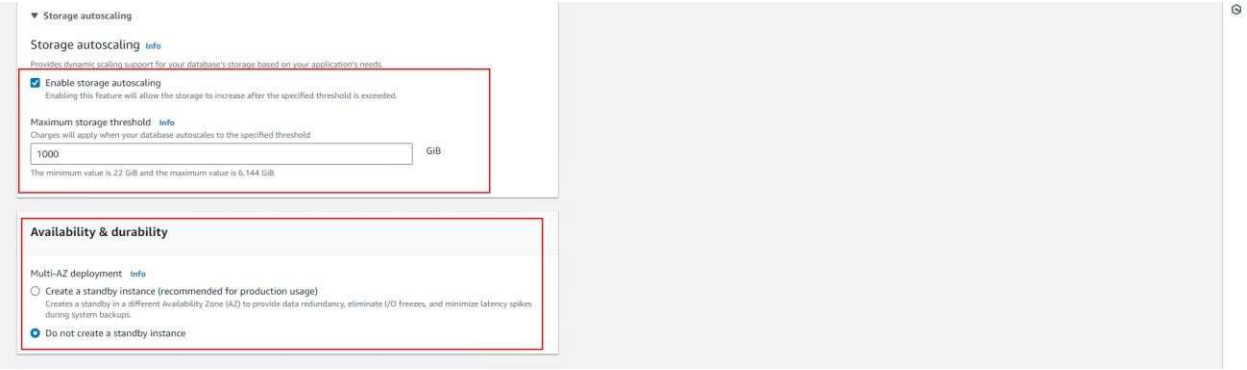
As shown in the image below, our desired **database has been created**.
Now, click on your database to configure its **advanced settings**.



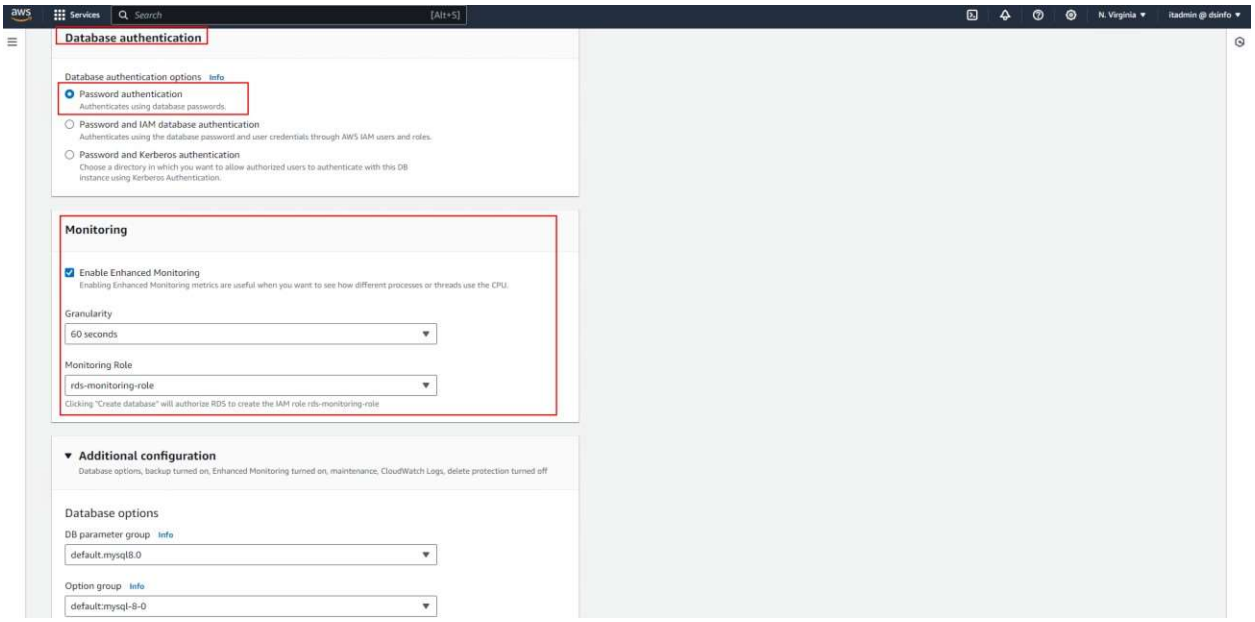
In this section, you can specify the **amount of resources** required for the database, the **type of storage**, and its **size**.



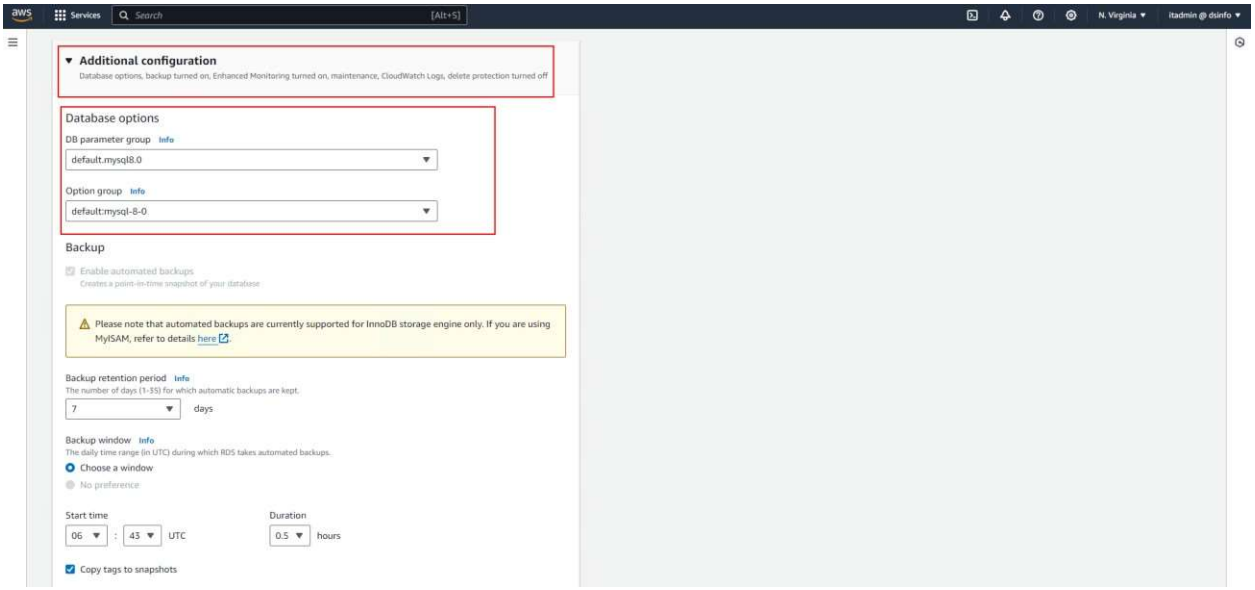
In this section, you can define the **maximum size** of the database and specify whether you want your database to be **replicated across multiple Availability Zones** or not.



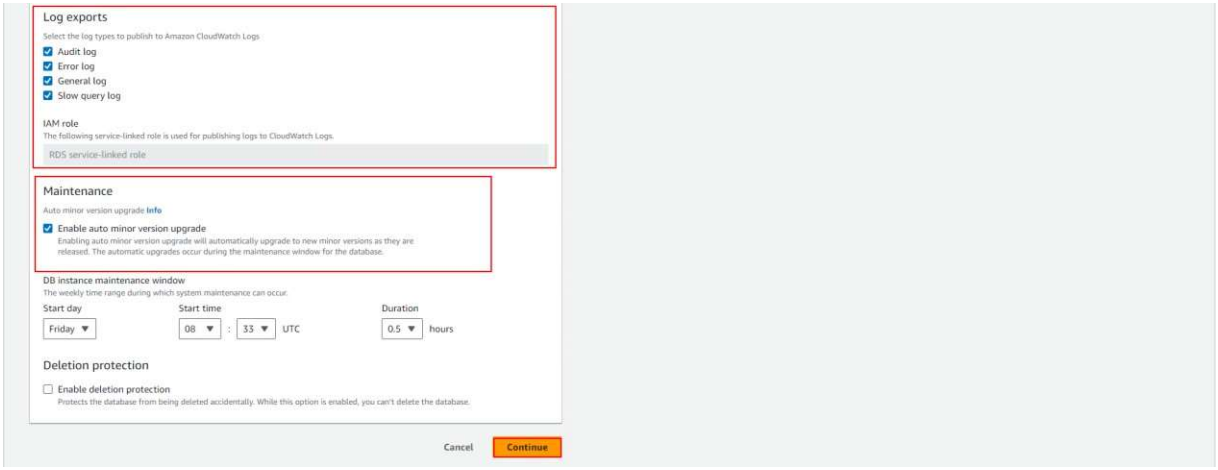
In this section, you can configure **monitoring** for the database.



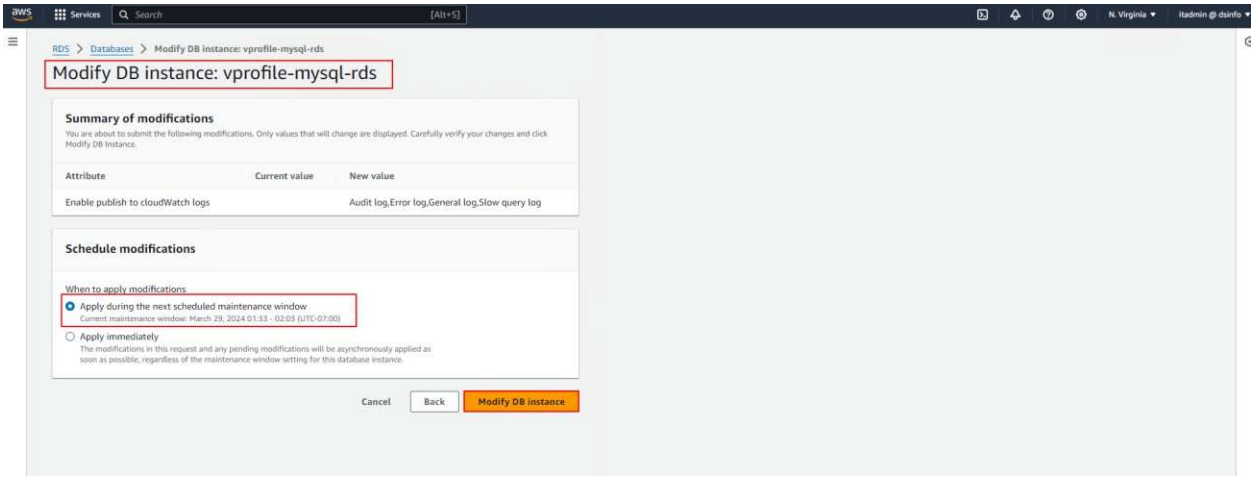
In this section, you can specify the **version** of your database.



In this section, you need to configure the **logging settings**, and then click the **Continue** button.



And at this stage, you need to click on the **Modify DB Instance** button.



By selecting the database and using the **Actions** menu, you can perform a variety of tasks on your database.