

MASTERING AWS CLOUD

FROM BEGINNER TO PRO



Fariborz Fallahzadeh

Instructor: Fariborz Fallahzadeh

Email Address:fariborz.fallahzadeh@gmail.com

Table Of Content

Introduction to Data Center Structure

What is Cloud Computing?

Benefits of Cloud Computing

Types of Cloud Computing

Introduction to AWS Free Tier

How to Create an AWS Free Tier Account

How to Create an IAM User in AWS

How to Set Up CloudWatch in AWS

Introduction to AWS Regions and Availability Zones

Introduction to Availability Zones

Introduction to AWS EC2

How to Create an EC2 Instance

How to Create a Key Pair

How to Create a Security Group

Deploying a Web Server on an EC2 Instance

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

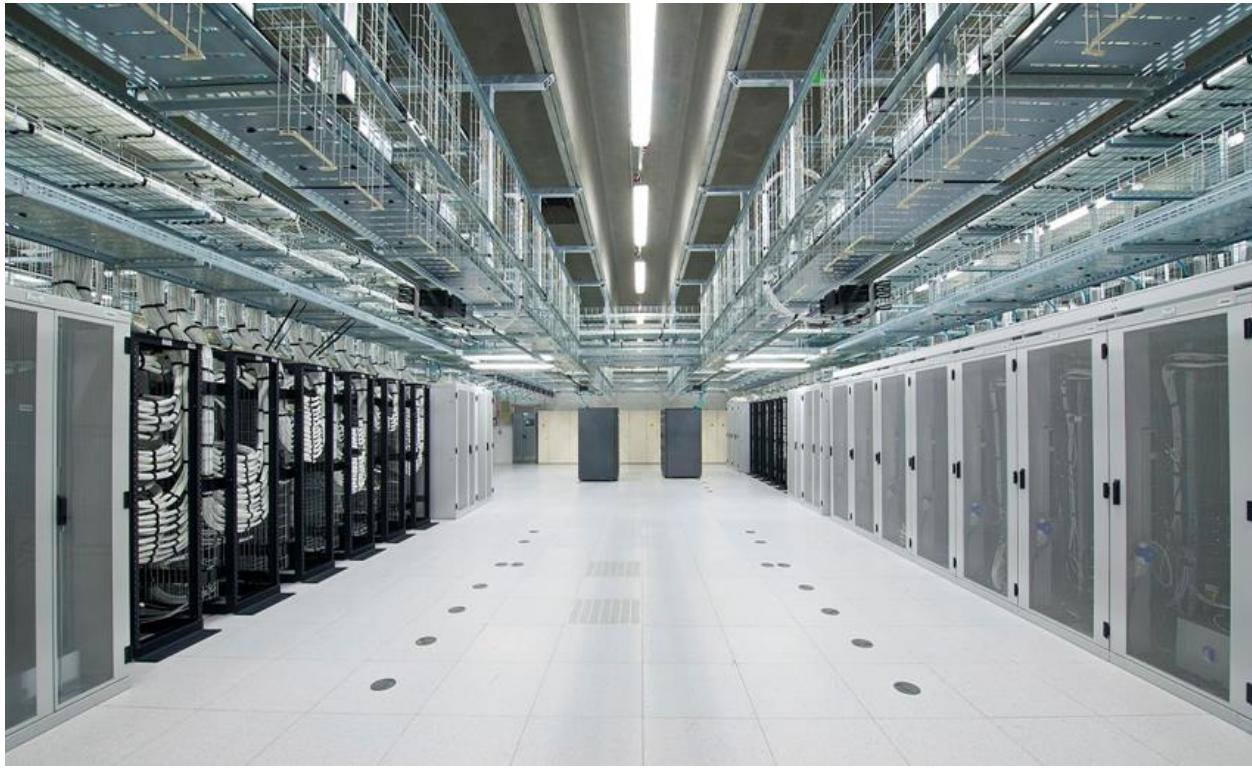
Introduction to Amazon S3

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.

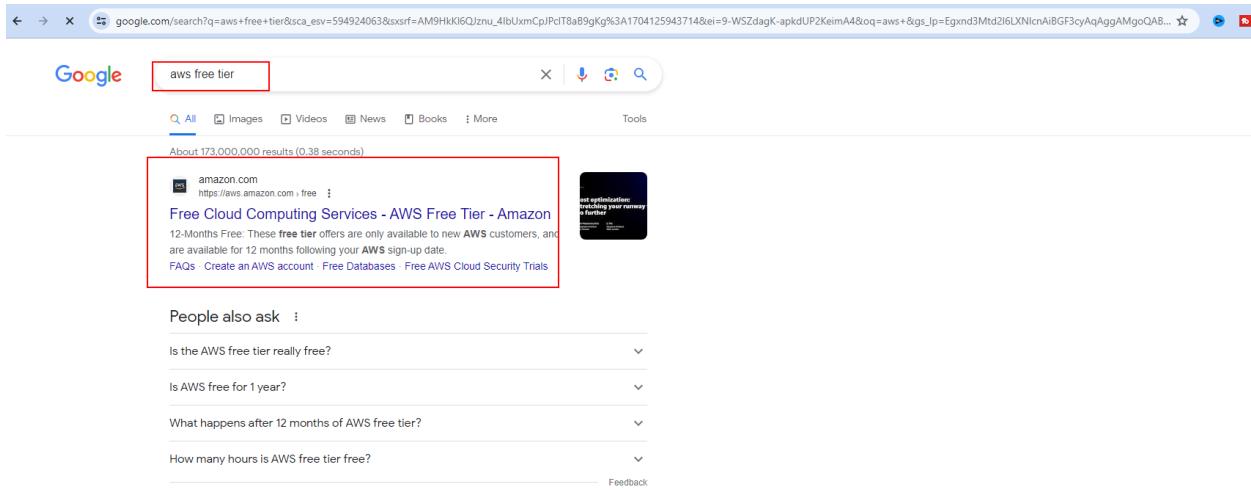
AWS is one of the largest cloud providers in the world today, holding nearly 46% of the market share.

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.

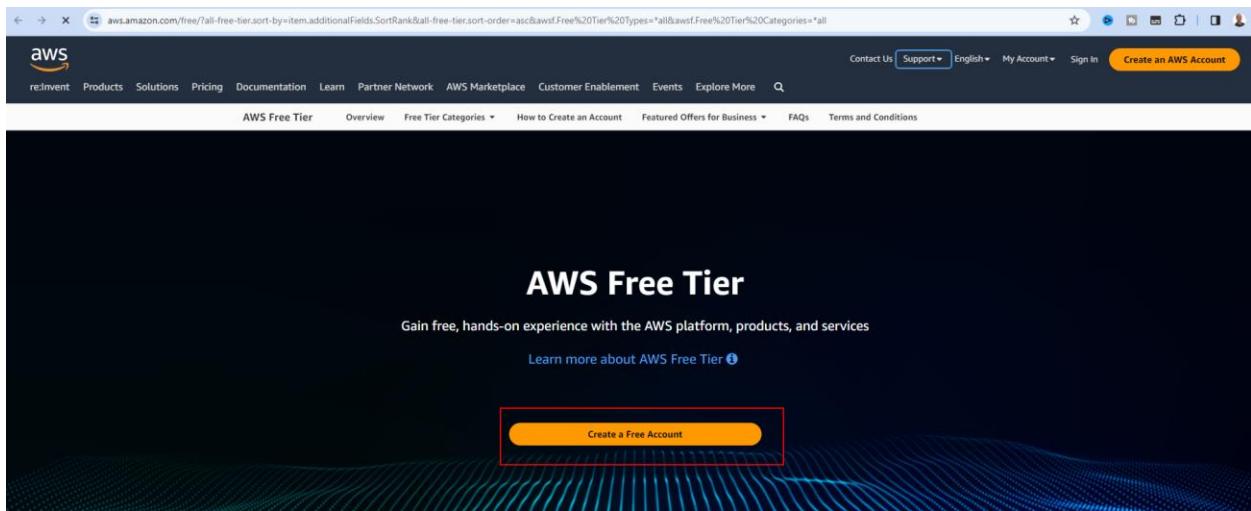
Finally, you can configure a certificate for your domain within AWS using the **ACM (AWS Certificate Manager)** service. This allows you to use **HTTPS** for secure access to your web services.

How to Create an AWS Free Tier Account

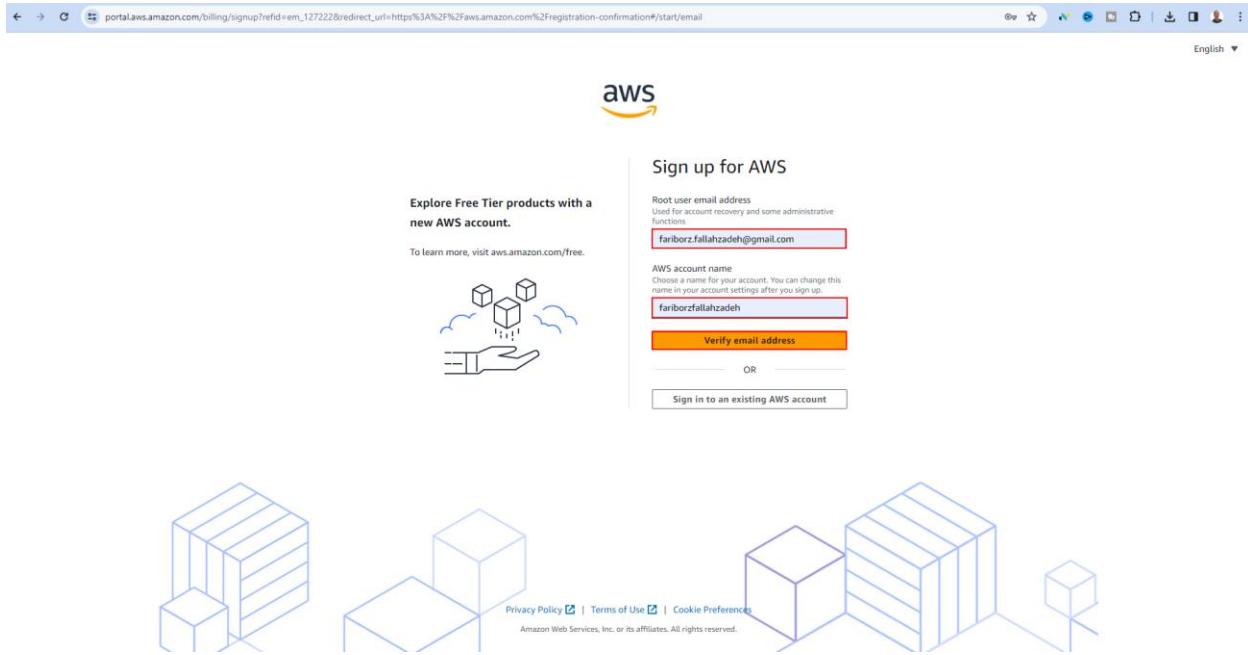
Search for “AWS Free Tier” on Google and then enter the Amazon website from the search results.



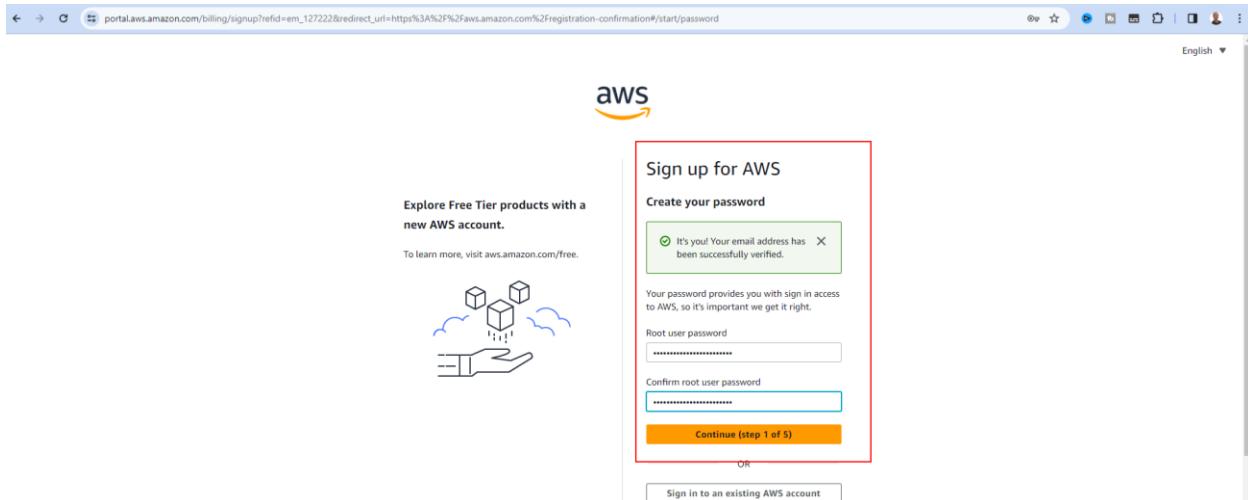
At this stage, click on the **Create Free Account** button.



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**.

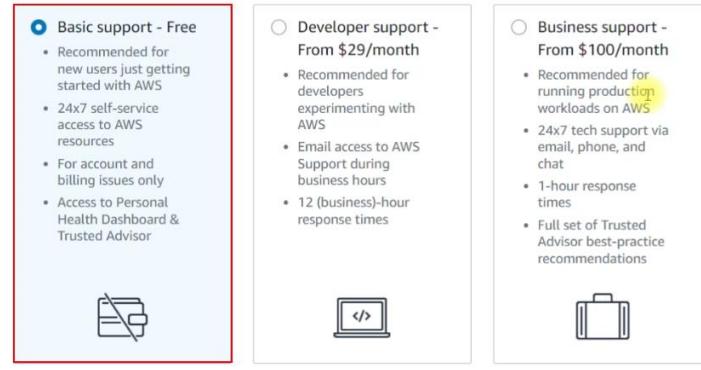


Next, you need to enter your **billing information** and **credit or debit card details**.

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.

Select a support plan

Choose a support plan for your business or personal account. [Compare plans and pricing examples](#)
 You can change your plan anytime in the AWS Management Console.



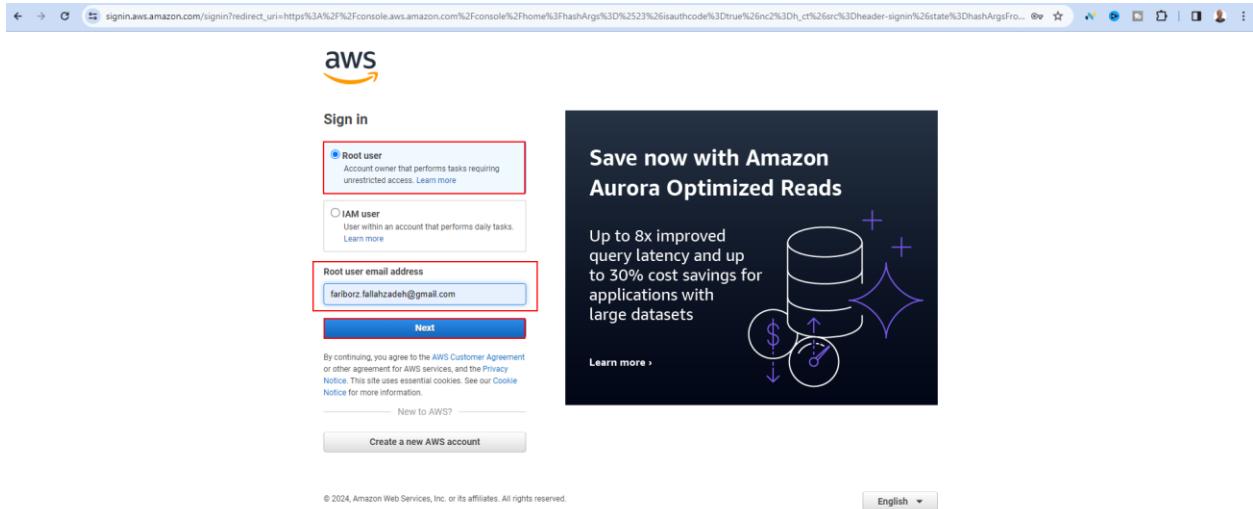
Finally, click on the **Complete Sign Up** button to finish the account creation process.

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



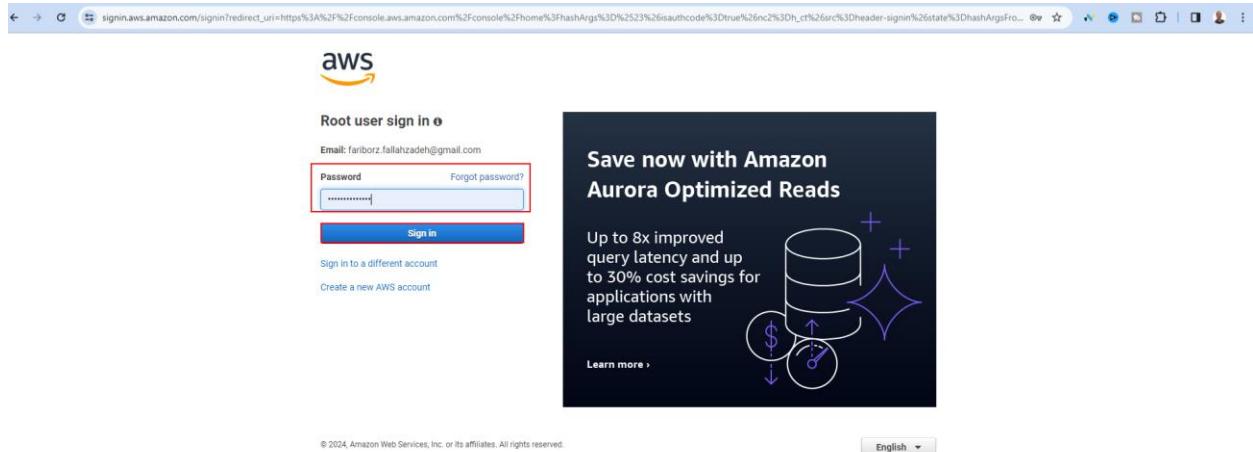
Then, click the button at the top to enter the **AWS Management Console**.

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.

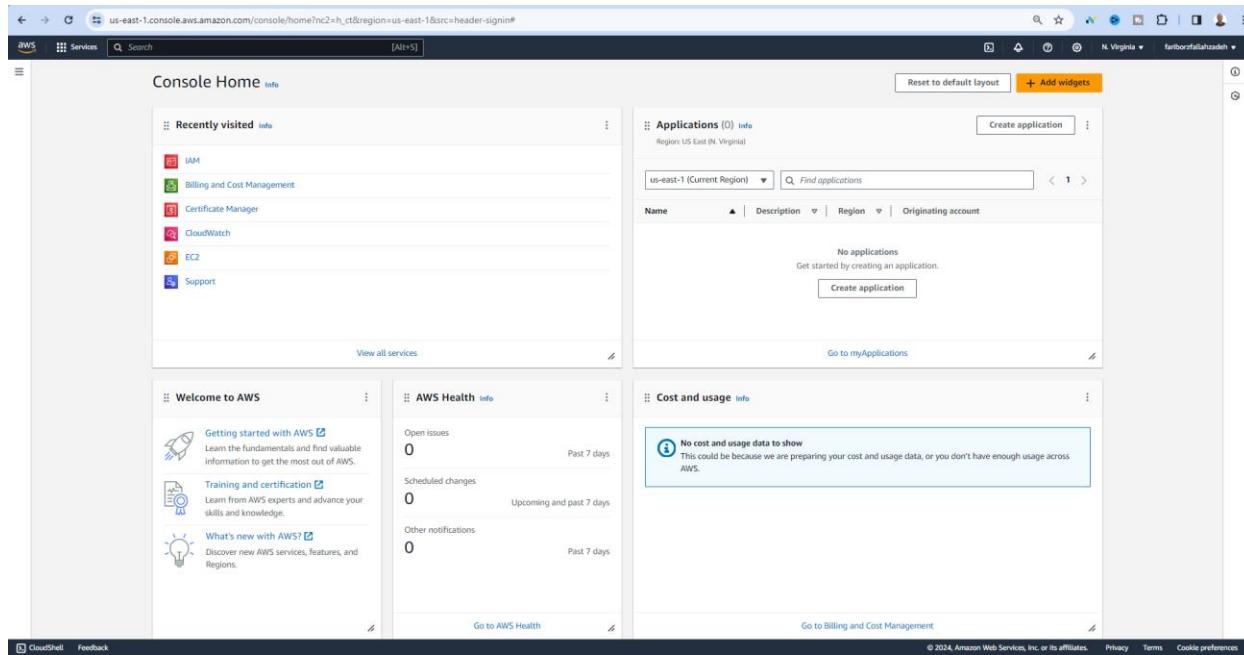


After entering the **Root Username**, click the **Next** button.

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.

The screenshot shows the AWS Management Console search results for the term 'ec2'. The search bar at the top contains 'ec2'. The left sidebar has a 'Services' section with 'EC2' highlighted. The main search results list includes 'EC2' (Virtual Servers in the Cloud), 'EC2 Image Builder' (A managed service to automate build, customize and deploy OS images), 'Recycle Bin' (Protect resources from accidental deletion), and 'Amazon Inspector' (Continual vulnerability management at scale). Below this is a 'Features' section with 'EC2 Instances' listed under 'CloudWatch feature'. The right side of the screen shows the 'Applications' page with no applications listed, and the 'Cost and usage' page with a message indicating 'No cost and usage data to show'.

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

The screenshot shows the AWS EC2 Dashboard for the US East (N. Virginia) Region. The left sidebar contains navigation links for EC2 Global View, Events, Console-to-Code, Instances, Images, Elastic Block Store, Network & Security, Load Balancing, and more. The main content area displays the following statistics:

Category	Value
Instances (running)	0
Elastic IPs	0
Load balancers	0
Snapshots	0
Auto Scaling Groups	0
Instances	0
Placement groups	0
Dedicated Hosts	0
Key pairs	0
Security groups	0
Volumes	1

Below the statistics, there are sections for Launch instance, Service health, Zones, and Scheduled events. The Service health section shows the region as US East (N. Virginia). The Zones section lists six availability zones (us-east-1a through us-east-1f) with their respective Zone IDs. The Scheduled events section indicates "No scheduled events". On the right side, there are sections for EC2 Free Tier info, Account attributes (including Default VPC and Settings), and Explore AWS.

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.

The screenshot shows the AWS EC2 Dashboard for the US East (N. Virginia) Region. The left sidebar includes sections for Instances, Images, Elastic Block Store, Network & Security, and Load Balancing. The main area displays resource statistics and management tools like Launch instance, Instance alarms, Scheduled events, and Migrate a server. A red box highlights the 'Support Center' button in the top right corner of the dashboard. An overlay window titled 'Support Center' is open, listing options such as 'Expert Help', 'Documentation', 'Training', and 'Getting Started Resource Center'. The bottom right of the dashboard includes links for 'View all AWS Free Tier offers', 'Default VPC', 'Settings', and 'Explore AWS'.

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 screenshot shows the AWS EC2 console interface. In the top navigation bar, the search bar contains the text 'IAM'. Below the search bar, the sidebar menu is visible, showing various AWS services like EC2 Dashboard, Instances, Images, and Network & Security. The main content area displays search results for 'IAM'. The first result, 'IAM', is highlighted with a red box. Other results listed are IAM Identity Center, Resource Access Manager, and AWS App Mesh. Below the search results, there are sections for 'Features' (Groups, Roles, Policies, Roles Anywhere) and 'Account attributes' (Default VPC, Settings). On the right side of the screen, there are sections for 'EC2 Free Tier Info' (0 offers in use), 'Exceeds free tier' (0 offers exceeded), and 'Explore AWS' (10 Things You Can Do Today to Reduce AWS Costs).

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.

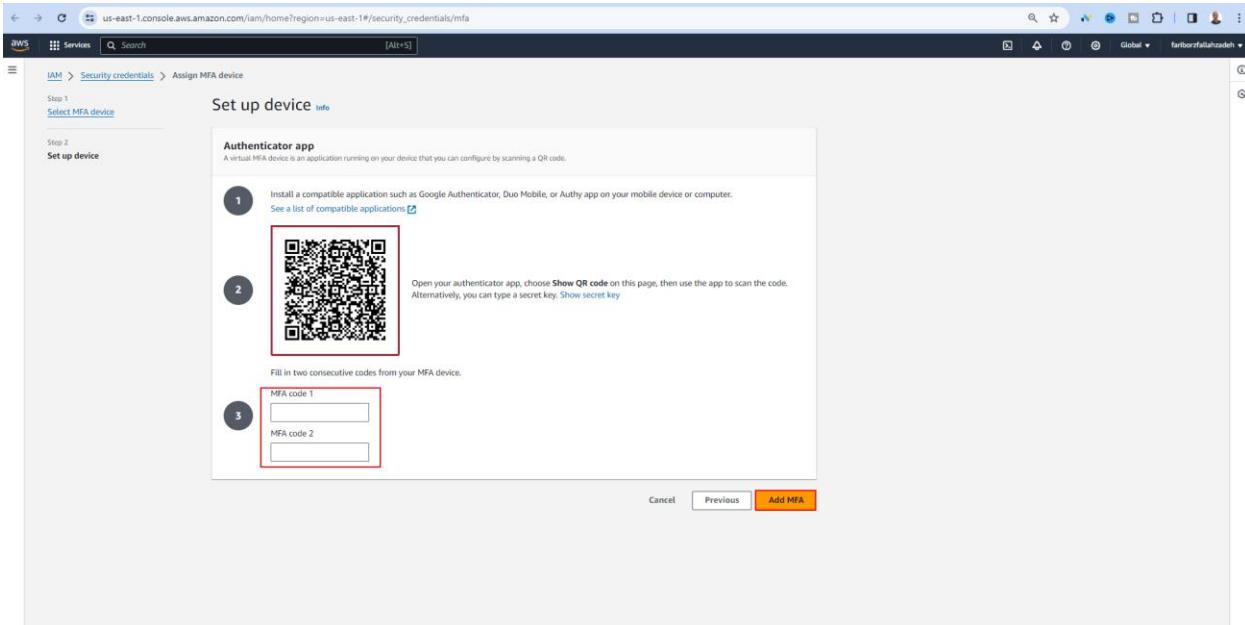
First, click on the **Add MFA** button.

The screenshot shows the AWS IAM Dashboard. On the left sidebar, under 'Access management', there is a red box around the 'Add MFA' button in the 'Security recommendations' section. The main area displays 'IAM resources' with counts for User groups (0), Users (0), Roles (2), Policies (0), and Identity providers (0). Below this is a 'What's new' section with a red box around the first item: 'IAM Access Analyzer now simplifies inspecting unused access to guide you toward least privilege.' At the bottom right of the dashboard, there is a 'Quick Links' section with a red box around the 'My security credentials' link.

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.

The screenshot shows the 'Select MFA device' step of the 'Assign MFA device' wizard. It is Step 1 of 2. The 'MFA device name' field contains 'Myphone'. Under 'MFA device', the 'Authenticator app' option is selected, indicated by a red box. The other options, 'Security Key' and 'Hardware TOTP token', are shown below. At the bottom right, there is a 'Cancel' button and a red box around 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.

The screenshot shows the AWS IAM Dashboard. On the left, the navigation menu includes 'Identity and Access Management (IAM)', 'Dashboard', 'Access management' (with 'User groups', 'Users', 'Roles', 'Policies', 'Identity providers', 'Account settings'), 'Access reports' (with 'Access Analyzer', 'External access', 'Unused access', 'Analyzer settings', 'Credential report', 'Organization activity', 'Service control policies (SCPs)'), and 'Related consoles' (with 'IAM Identity Center' and 'AWS Organizations'). The main content area is titled 'IAM Dashboard' and contains a 'Security recommendations' section with two items: 'Root user has MFA' (having multi-factor authentication (MFA) for the root user improves security for this account) and 'Root user has no active access keys' (using access keys attached to an IAM user instead of the root user improves security). Below this is an 'IAM resources' section showing 0 User groups, 0 Users, 2 Roles, 0 Policies, and 0 Identity providers. A 'What's new' section lists recent updates from the IAM Access Analyzer. On the right, there are sections for 'AWS Account' (Account ID: 095418366137, Account Alias: Create, Sign-in URL: https://095418366137.sigin.aws.amazon.com/console), 'Quick Links' (My security credentials, Policy simulator, Additional information), and 'Tools' (Policy simulator).

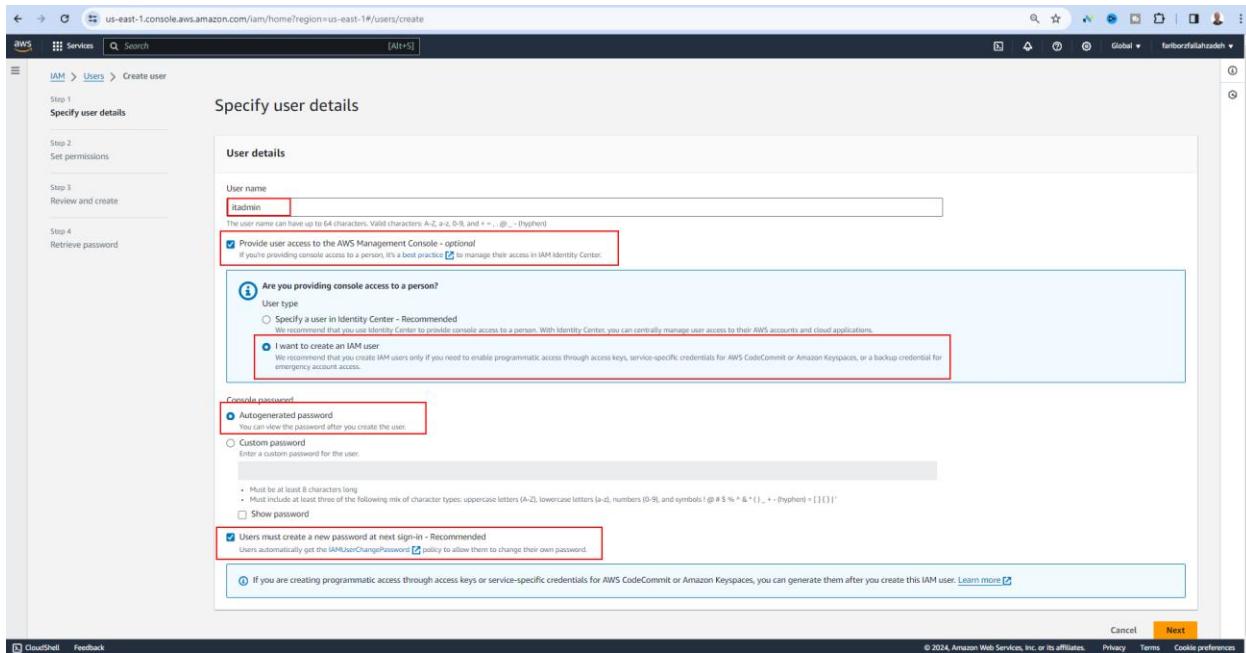
At this stage, to create an IAM user, click on the **Users** link and then click the **Add Users** button.

The screenshot shows the AWS IAM Users page. The left navigation menu is identical to the previous dashboard. The main content area is titled 'Users (0) info' and contains a table header for 'User name', 'Path', 'Group', 'Last activity', 'MFA', 'Password age', 'Console last sign-in', 'Access key ID', 'Active key age', and 'Access key last use'. Below the table, it says 'No resources to display'. At the top right of the table area, there is a red-bordered 'Create user' 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.

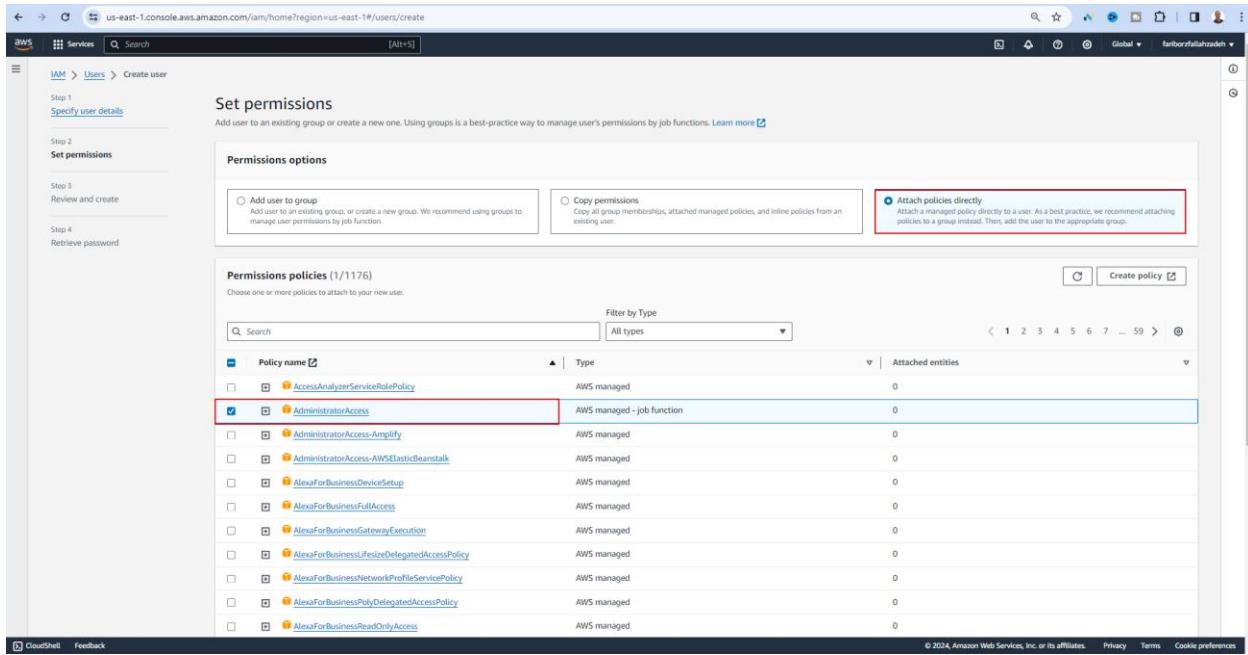
Next, select "**I want to create an IAM user**", then choose "**Autogenerated password**", and also check the box "**User must create a new password at next sign-in**".

Finally, click the **Next** button.



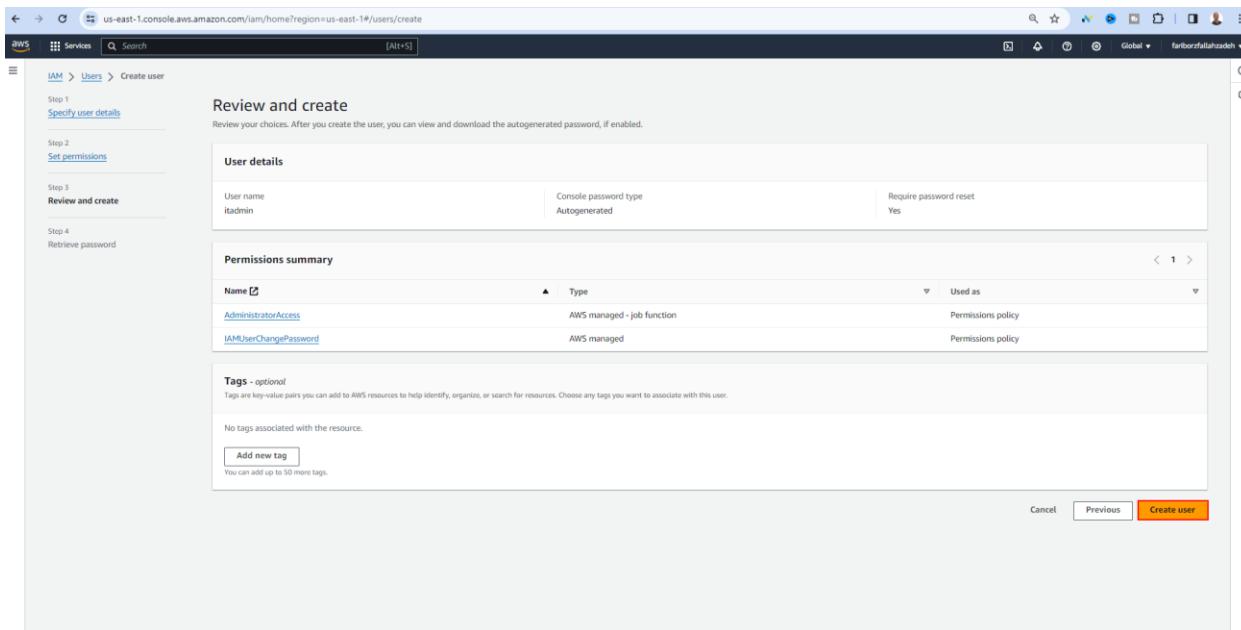
By default, this user has no permissions. To assign permissions, you can select the **Attach policies directly** option and then grant **AdministratorAccess** to the user.

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



The screenshot shows the AWS IAM 'Create user' wizard at Step 2: Set permissions. The 'AdministratorAccess' policy is selected and highlighted with a red box. Other policies listed include AccessAnalyzerServiceRolePolicy, AdministratorAccess-Amplify, AdministratorAccess-AWSLambda, AlexaForBusinessDeviceSetup, AlexaForBusinessFullAccess, AlexaForBusinessGatewayExecution, AlexaForBusinessIotEdgeDelegatedAccessPolicy, AlexaForBusinessNetworkProfileServicePolicy, AlexaForBusinessPolicyDelegatedAccessPolicy, and AlexaForBusinessReadOnlyAccess. The 'Attach policies directly' option is also highlighted with a red box.

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.

The screenshot shows the AWS IAM Users page. At the top, a green banner displays the message "User created successfully". Below the banner, the page title is "Users (1) Info". A sub-header states, "An IAM user is an identity with long-term credentials that is used to interact with AWS in an account." A search bar labeled "Search" is present. The main content is a table titled "Users (1) Info" with one row. The row contains the following data:

User name	Path	Group	Last activity	MFA	Password age	Console last sign-in	Access key ID	Active key age	Access key last use
fariborz.fallahzadeh	/	0							

The "User name" column for the first row is highlighted with a red box. The page footer includes links for CloudShell, Feedback, and various AWS services like IAM Identity Center and AWS Organizations, along with copyright information and cookie preferences.

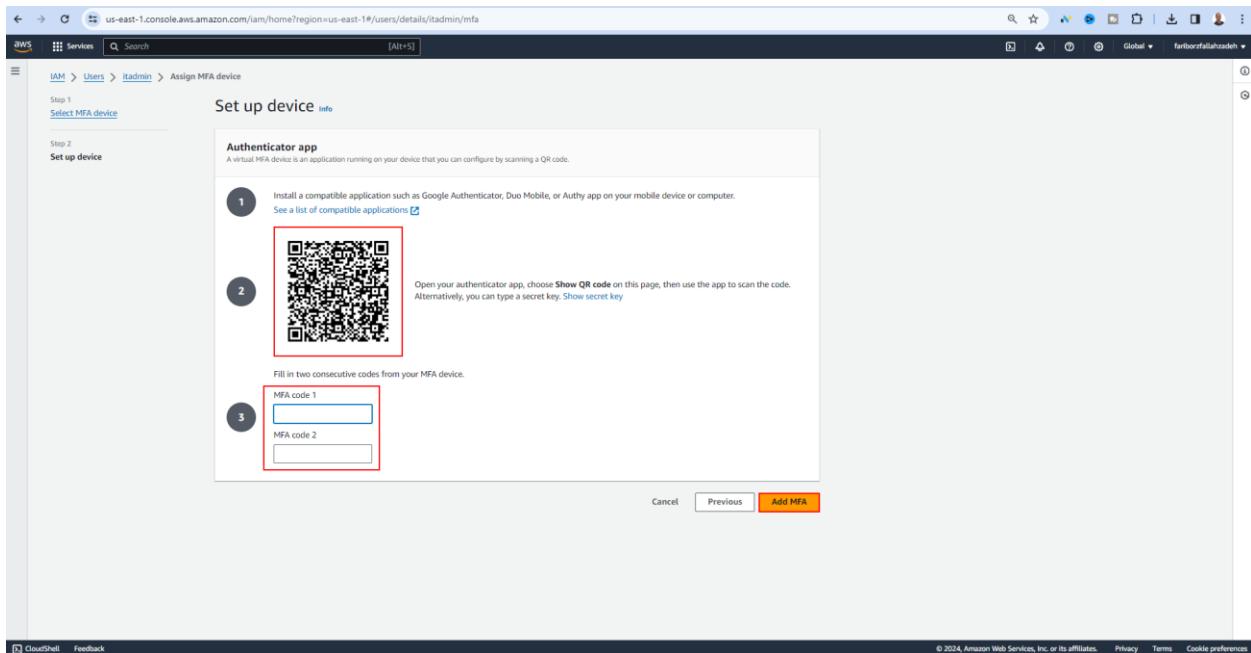
Next, click on the **Security Credentials** section and then click on the **Assign MFA Device** button.

The screenshot shows the AWS IAM User Details page for a user named 'itadmin'. The 'Security credentials' tab is highlighted with a red box. Under the 'Multi-factor authentication (MFA)' section, there is a button labeled 'Assign MFA device' which is also highlighted with a red box.

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.

The screenshot shows the 'Select MFA device' step in the AWS IAM User Details process. The 'Authenticator app' option is selected and highlighted with a red box. The 'Next' button at the bottom right is also highlighted with a red box.

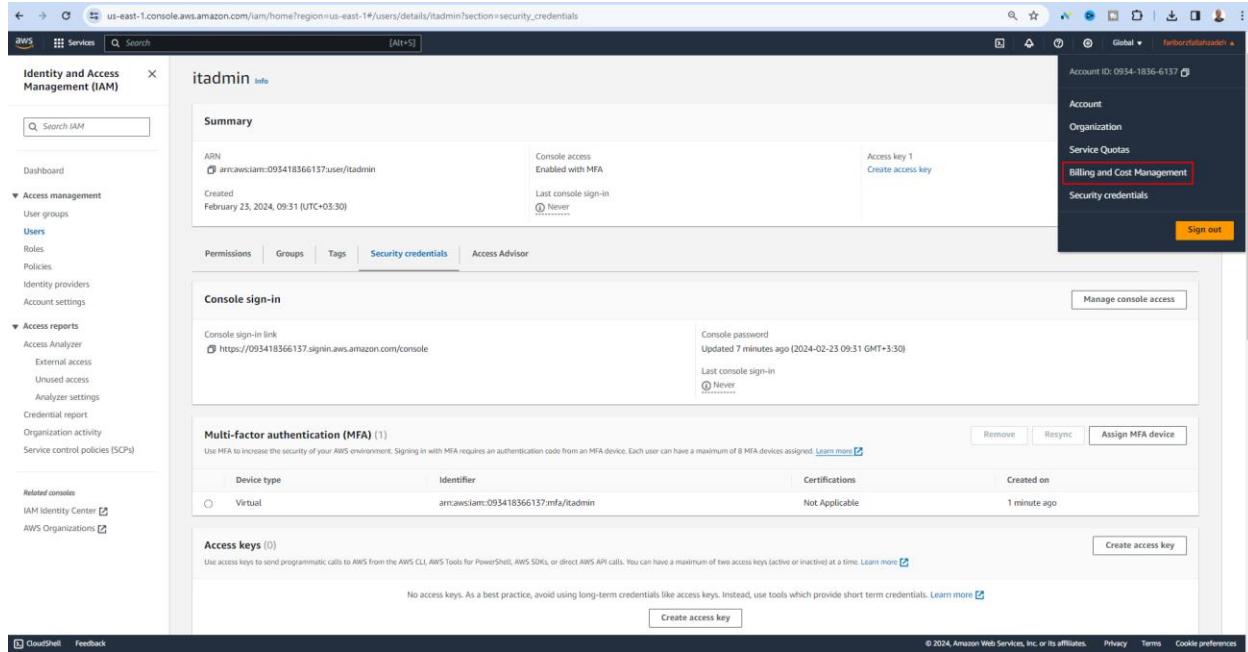
Then, go to your phone, open the **Google Authenticator** app, tap the + button, and scan the **QR code** provided by AWS. Enter the two **MFA codes** generated by Google Authenticator in the respective fields, and finally, click the **Add MFA** button.



How to Set Up CloudWatch in AWS

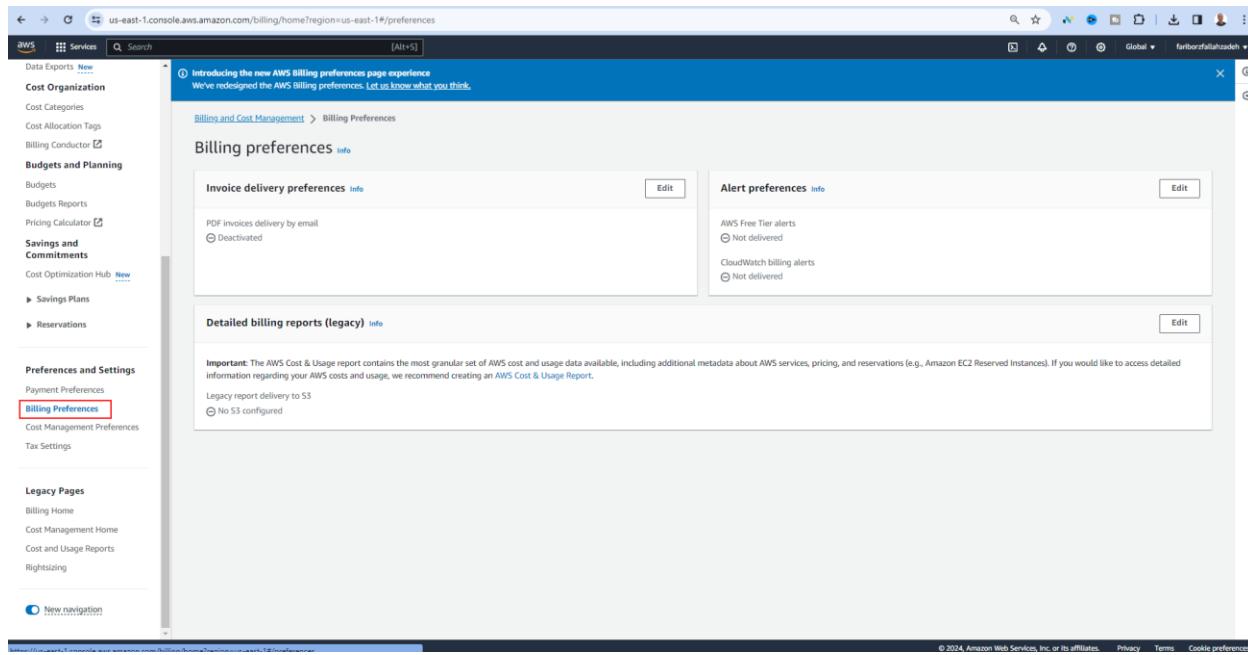
To monitor the usage of services in AWS and control costs, you need to configure a few options under the **Billing and Cost Management** section. This will allow you to set up a **Billing Alarm**, which will notify you when usage exceeds the specified threshold. Additionally, you should delete any created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

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



The screenshot shows the AWS IAM user details page for 'itadmin'. The 'Security credentials' tab is selected. On the right side of the page, there is a sidebar with several links: 'Account', 'Organization', 'Service Quotas', 'Billing and Cost Management' (which is highlighted with a red box), and 'Security credentials'. The main content area displays user information like ARN, access keys, and MFA details.

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



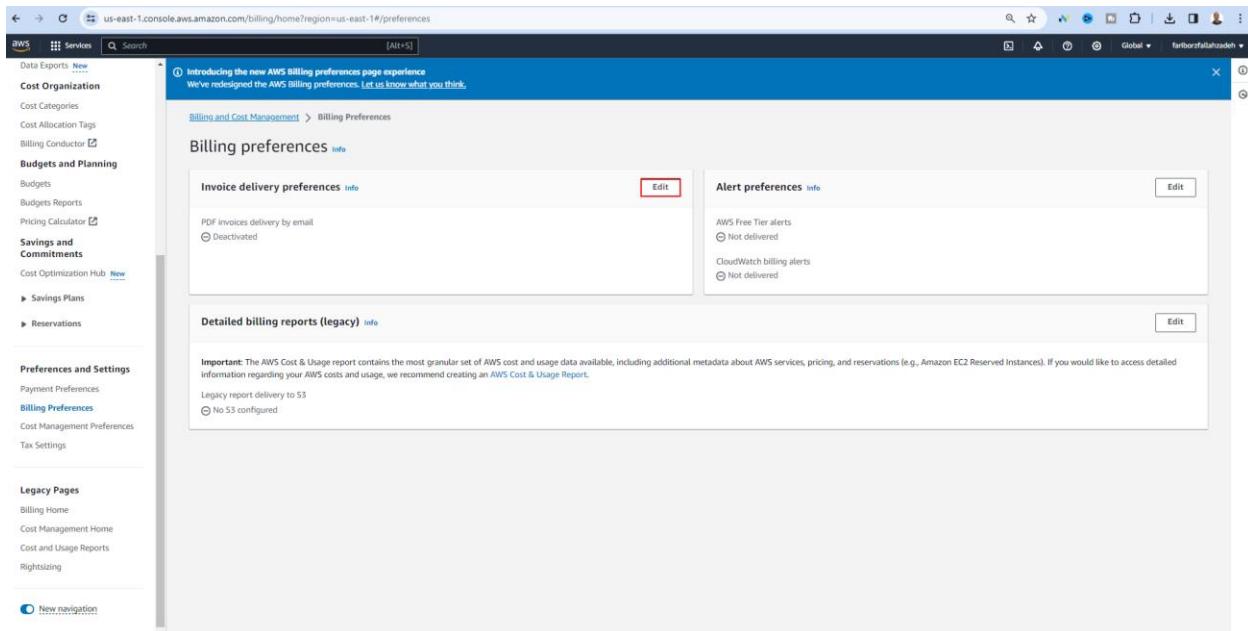
The screenshot shows the 'Billing preferences' page. The left sidebar has a 'Billing and Cost Management' section with a 'Billing Preferences' link, which is highlighted with a red box. The main content area contains sections for 'Invoice delivery preferences', 'Alert preferences', and 'Detailed billing reports (legacy)'. A note at the bottom of the detailed reports section states: 'Important: The AWS Cost & Usage report contains the most granular set of AWS cost and usage data available, including additional metadata about AWS services, pricing, and reservations (e.g., Amazon EC2 Reserved Instances). If you would like to access detailed information regarding your AWS costs and usage, we recommend creating an AWS Cost & Usage Report.'

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.

The screenshot shows the AWS Billing preferences page. On the left, there's a sidebar with navigation links like 'Cost Organization', 'Budgets and Planning', 'Preferences and Settings', and 'Legacy Pages'. The main content area has a header 'Introducing the new AWS Billing preferences page experience' with a note 'We've redesigned the AWS Billing preferences. Let us know what you think.' Below this, the 'Billing preferences' section is displayed. Under 'Invoice delivery preferences', there's a checkbox labeled 'PDF invoices delivered by email' which is checked and highlighted with a red border. There are also 'Update' and 'Cancel' buttons. To the right, under 'Alert preferences', there are sections for 'AWS Free Tier alerts' and 'CloudWatch billing alerts', both with 'Not delivered' radio buttons selected. A note at the bottom of this section says 'Important: The AWS Cost & Usage report contains the most granular set of AWS cost and usage data available, including additional metadata about AWS services, pricing, and reservations (e.g., Amazon EC2 Reserved Instances). If you would like to access detailed information regarding your AWS costs and usage, we recommend creating an AWS Cost & Usage Report.' At the very bottom of the page, there are links for 'ClearShell', 'Feedback', and copyright information: '© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences'.

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

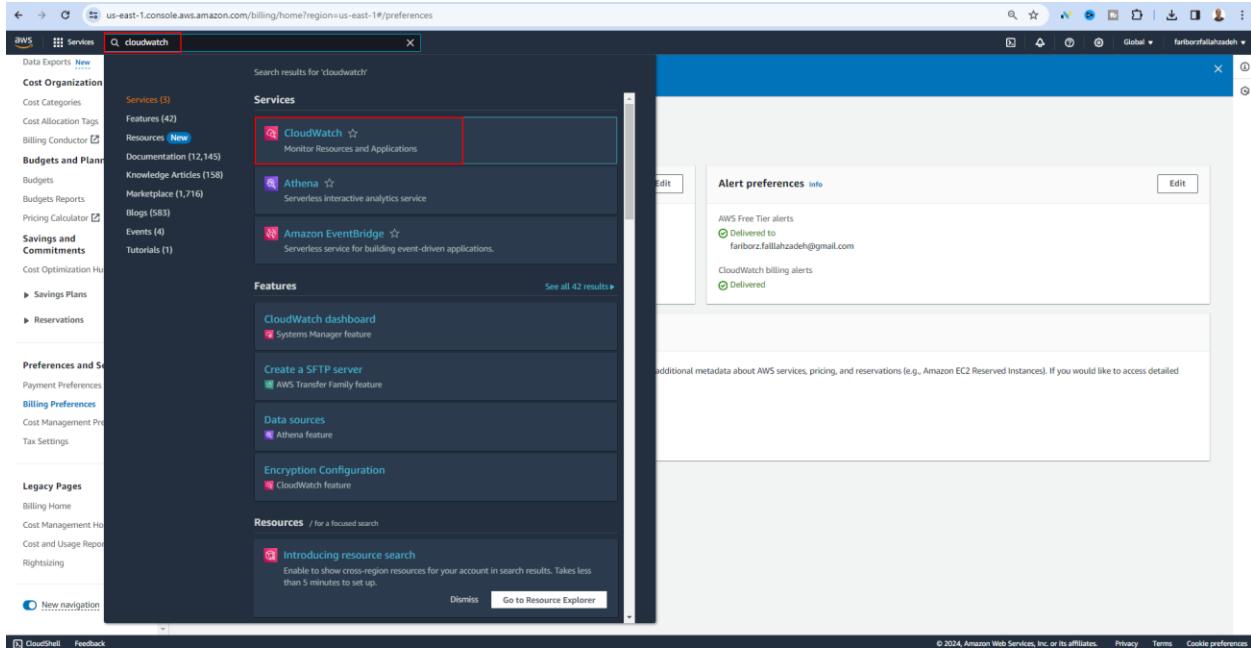
The screenshot shows the AWS Billing preferences page. On the left, there's a sidebar with navigation links like Cost Organization, Budgets, Savings and Commitments, and Preferences and Settings. The main content area has two main sections: 'Invoice delivery preferences' (which shows 'PDF invoices delivery by email' is activated) and 'Alert preferences'. Under 'Alert preferences', there are two options: 'AWS Free Tier alerts' (radio button selected) and 'CloudWatch billing alerts' (radio button selected). A message at the top says 'Your invoice delivery preferences were updated successfully.'

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.

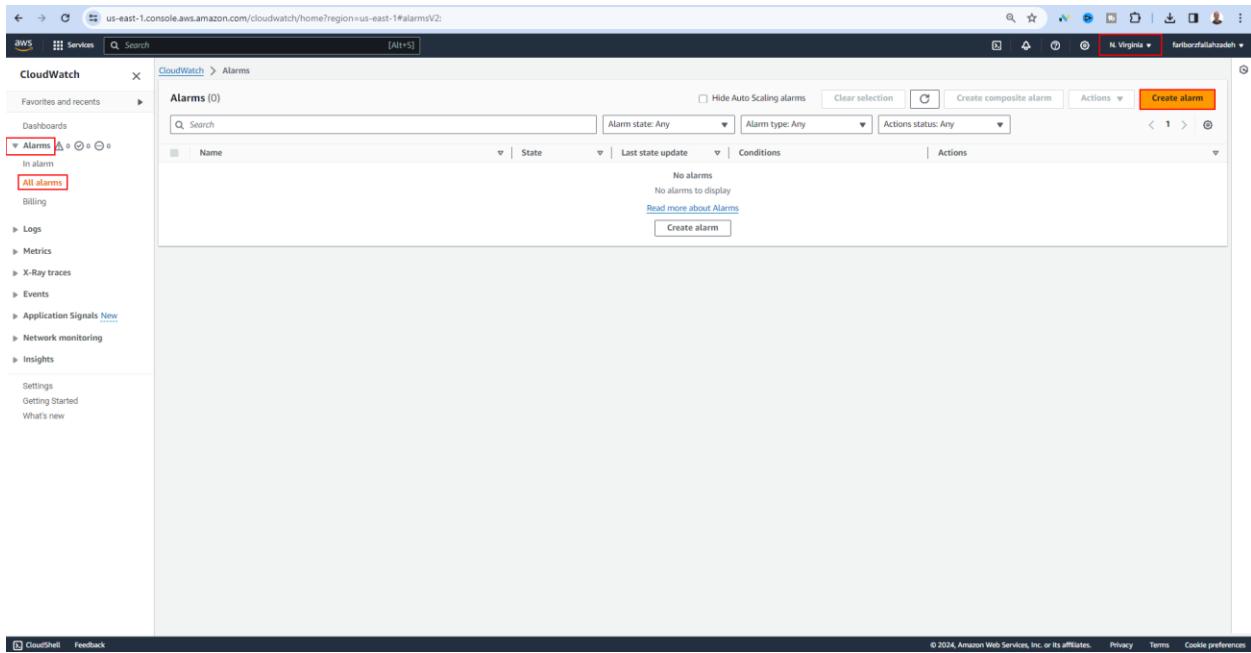
This screenshot is similar to the previous one but highlights the 'Alert preferences' section with a red border. It shows the same two options: 'Receive AWS Free Tier alerts' (selected) and 'Receive CloudWatch billing alerts' (selected). Below these options are input fields for an email address ('fariborz.fallahzadeh@gmail.com') and a note about CloudWatch alerts being不可逆的 (cannot be disabled). At the bottom right, there are 'Update' and 'Cancel' buttons.

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.

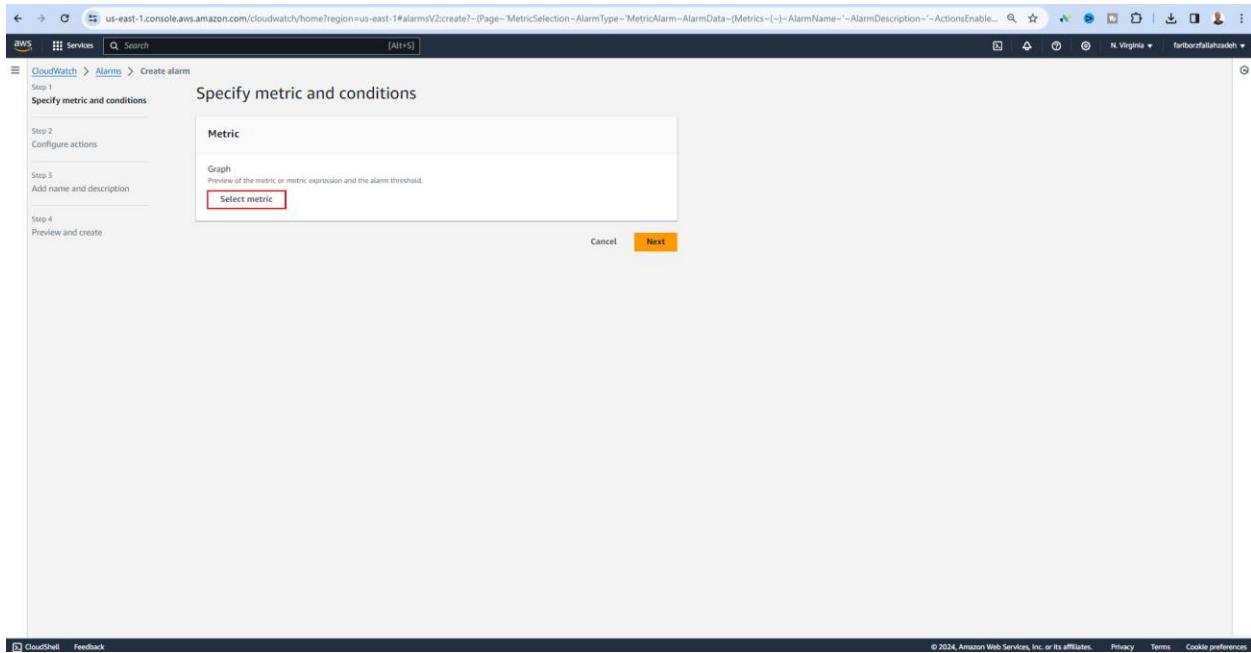


At this stage, click on the **All Alarms** link, then click the **Create Alarm** button.



The screenshot shows the AWS CloudWatch Alarms page. On the left, there's a navigation sidebar with various links like Dashboards, Alarms (which is selected and highlighted with a red border), In alarm, All alarms, Billing, Logs, Metrics, X-Ray traces, Events, Application Signals, Network monitoring, and Insights. The main content area is titled 'Alarms (0)' and contains a search bar and filters for Name, State, Last state update, and Conditions. A large orange 'Create alarm' button is located at the top right of this section. Below it, there's a message 'No alarms to display' and a 'Read more about Alarms' link. At the bottom of the page, there are links for CloudShell, Feedback, and copyright information.

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

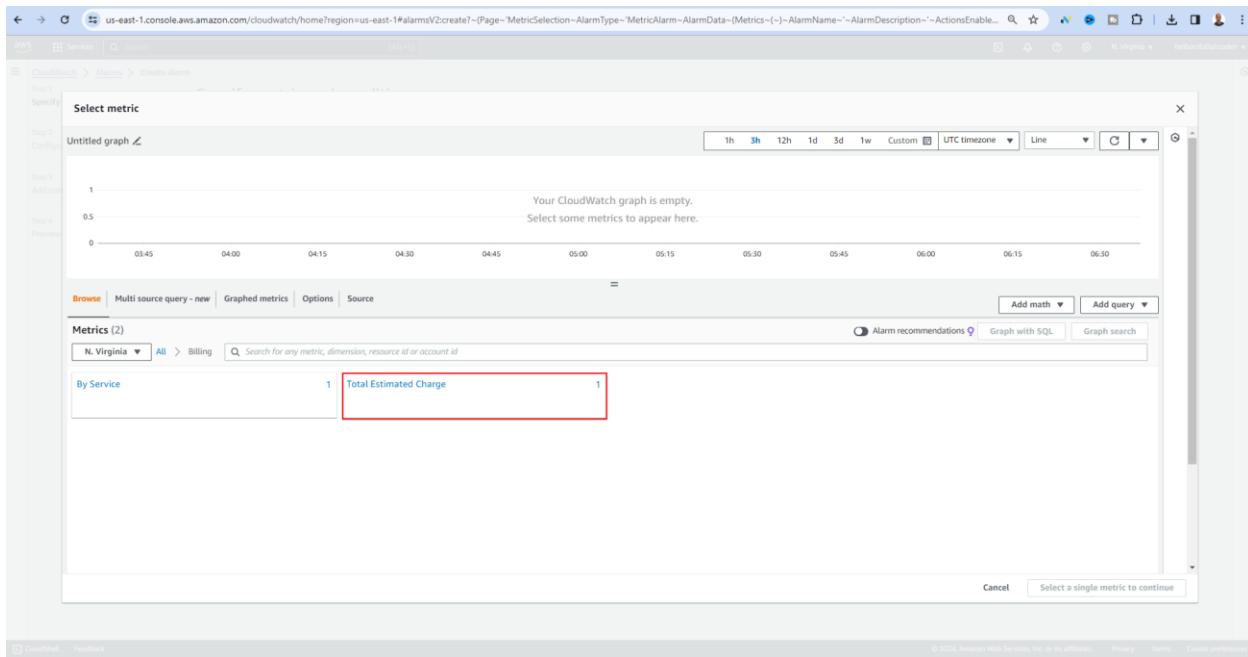


The screenshot shows the 'Specify metric and conditions' step of the 'Create alarm' wizard. On the left, a sidebar lists steps: Step 1 (Selected), Step 2 (Configure actions), Step 3 (Add name and description), and Step 4 (Preview and create). The main area has a title 'Specify metric and conditions' and a 'Metric' section containing a 'Graph' preview and a 'Select metric' button. At the bottom, there are 'Cancel' and 'Next' buttons. The page footer includes CloudShell, Feedback, and copyright information.

Click on the **Billing** link.

The screenshot shows the 'Select metric' dialog box from the AWS CloudWatch Metrics interface. The 'Metrics' section displays a list of 62 metrics. The 'Billing' metric is highlighted with a red box. Other metrics listed include 'Logs' (2), 'Usage' (2), and '58'. The dialog box also includes a graph area at the top showing an empty graph from 03:45 to 06:30, and a search bar at the bottom.

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



The screenshot shows the 'Select metric' dialog box from the AWS CloudWatch Metrics interface. The 'Metrics' section displays a list of 2 metrics. The 'Total Estimated Charge' metric is highlighted with a red box. Other metrics listed include 'By Service' (1) and 'Billing' (1). The dialog box also includes a graph area at the top showing an empty graph from 03:45 to 06:30, and a search bar at the bottom.

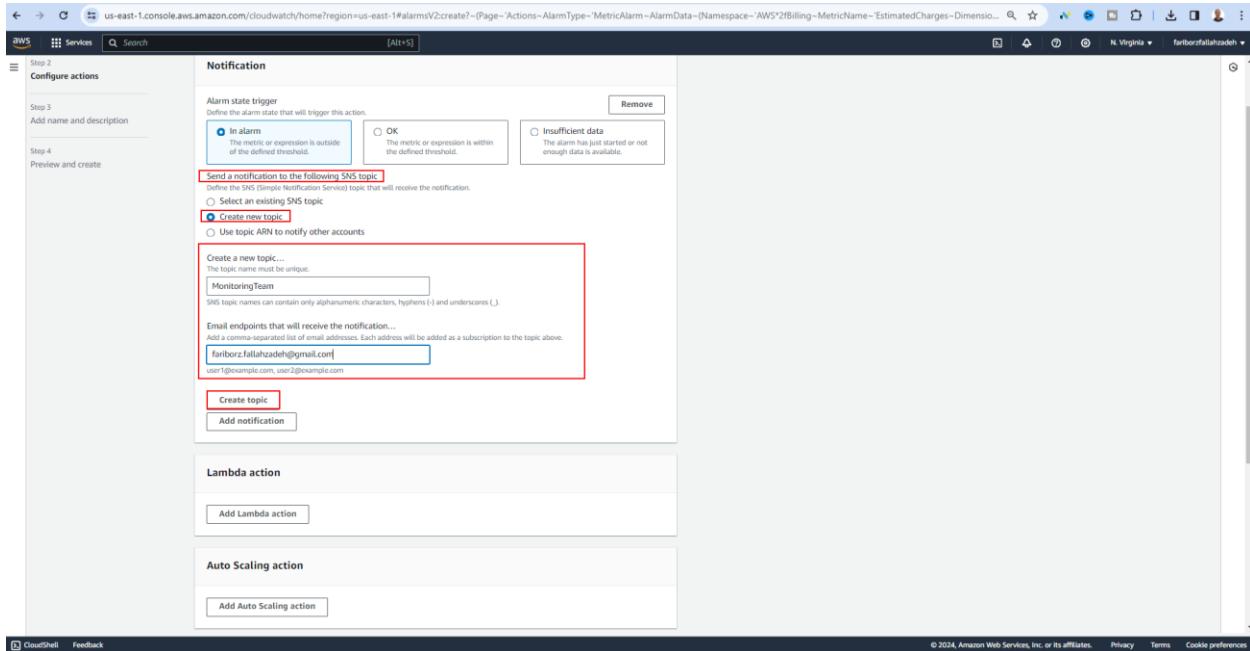
At this stage, select **USD Currency** and then click the **Select Metric** button.

The screenshot shows the 'Select metric' step of a CloudWatch alarm creation wizard. The 'Metrics (1)' section lists 'Currency 1/1' with 'EstimatedCharges' selected. Under 'Currency', 'USD' is highlighted with a red box. The 'Select metric' button is at the bottom right.

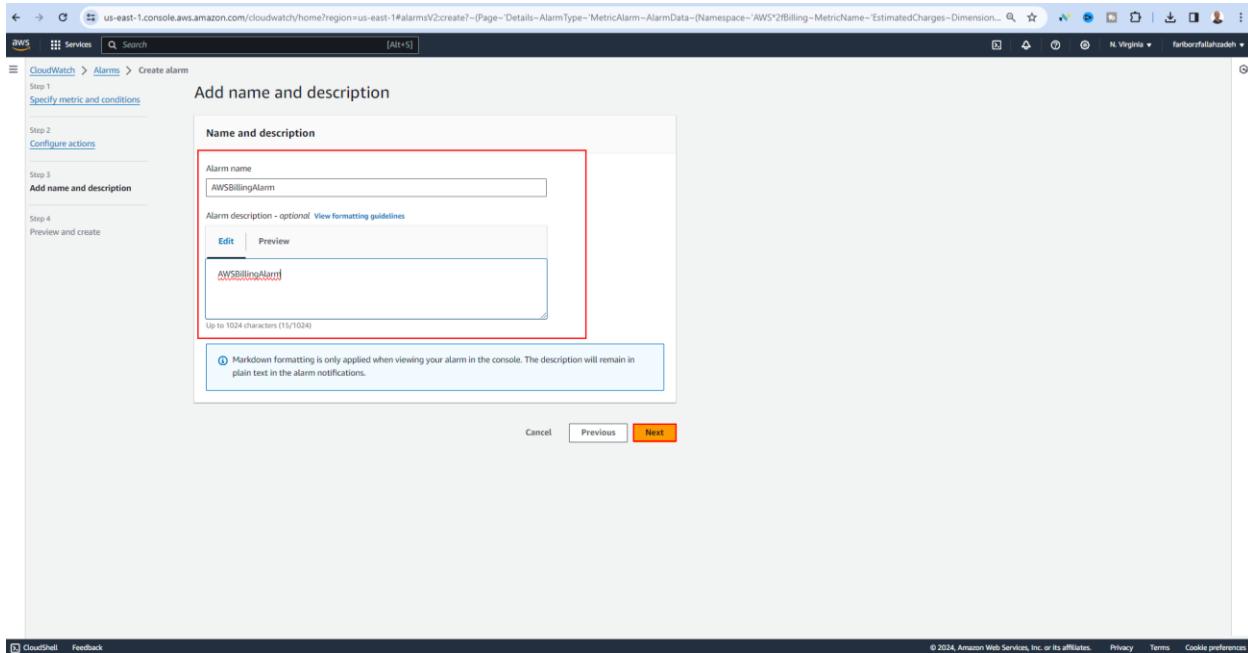
In the **Threshold value** section, set the value to **5 dollars** so that if the charges exceed 5 dollars, you will receive a notification via email. Then, click the **Next** button.

The screenshot shows the 'Step 4: Preview and create' screen. In the 'Conditions' section, under 'Threshold type', 'Static' is selected. The 'than...' section shows 'Greater > threshold' selected, and the threshold value input field contains '5'. The 'Next' button is at the bottom right.

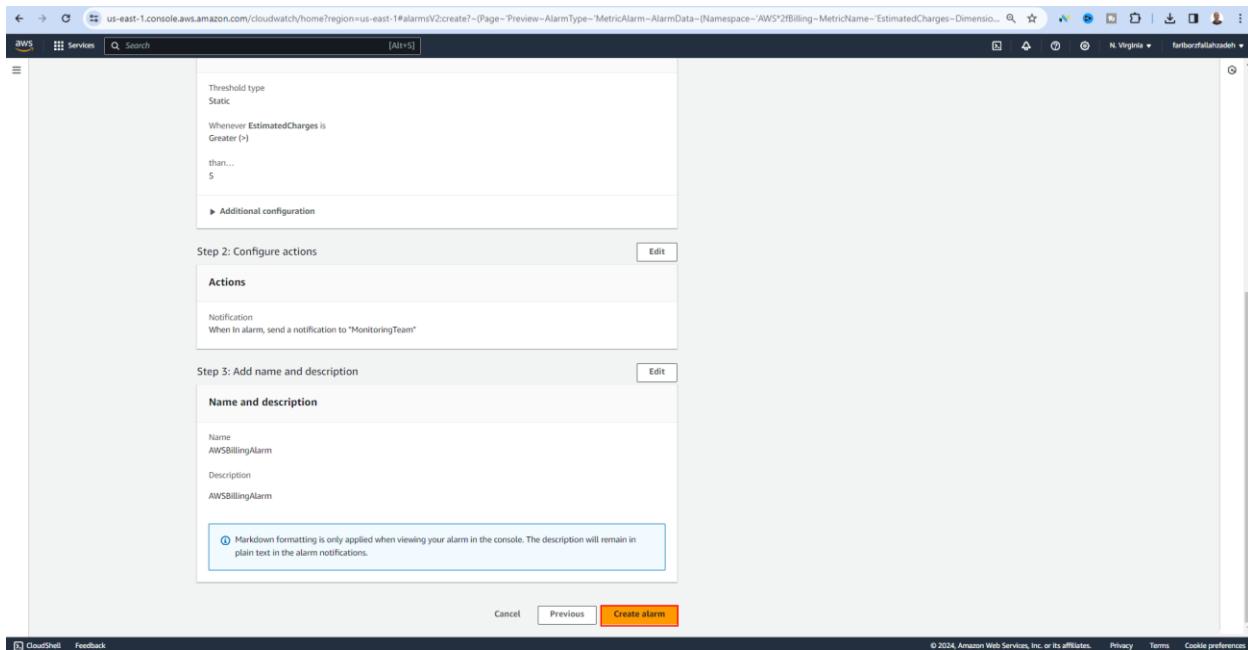
At this stage, you need to use a service called **SNS (Simple Notification Service)**. In this section, you will define an **SNS Topic**, which means you need to associate an email address with a topic. Define a name for the topic, then click the **Create Topic** button. Finally, click the **Next** button.



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.

The screenshot shows the AWS CloudWatch Alarms page. At the top, a green banner indicates "Successfully created alarm AWSBillingAlarm." Below this, a message says "Some subscriptions are pending confirmation" and "Amazon SNS doesn't send messages to an endpoint until the subscription is confirmed." The main table lists one alarm:

Name	State	Last state update	Conditions	Actions
AWSBillingAlarm	Insufficient data	2024-02-23 06:45:28	EstimatedCharges > 5 for 1 datapoints within 6 hours	Actions enabled Warning

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

The screenshot shows a Gmail inbox with 12,309 unread emails. An email from "AWS Notifications <no-reply@sns.amazonaws.com>" is selected. The subject is "AWS Notification - Subscription Confirmation". The email body contains the following text:

You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:093418366137:MonitoringTeam
To confirm this subscription, click or visit the link below (If this was in error no action is necessary).
[Confirm subscription](#)

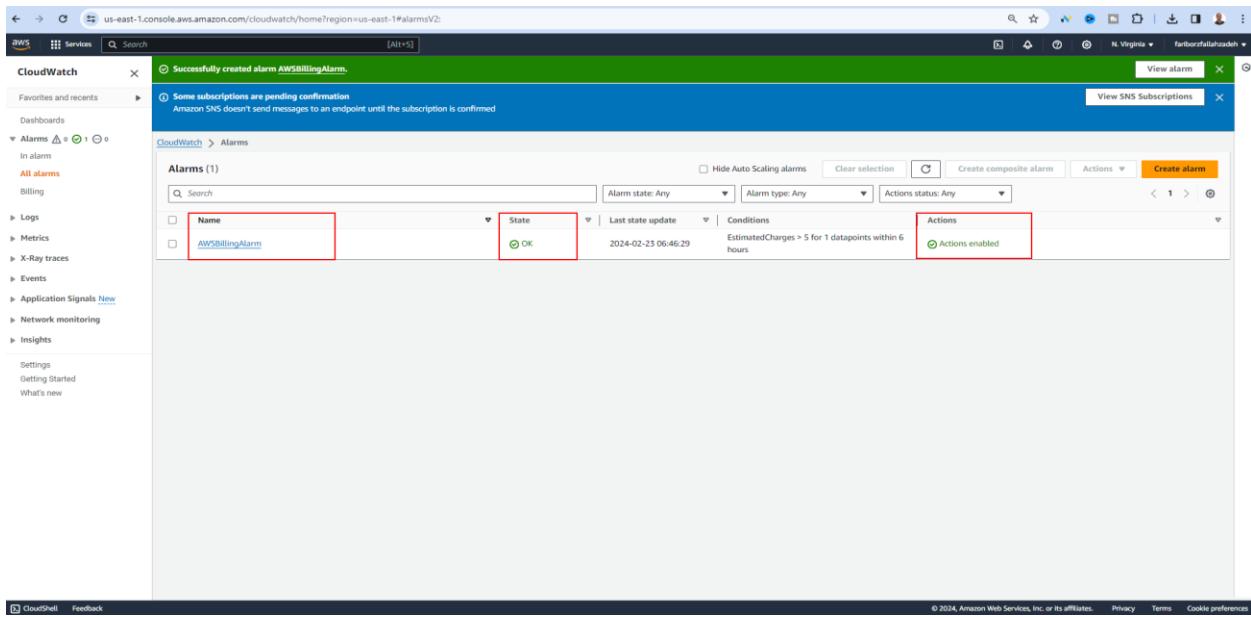
Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to [sns:dot:out](#).

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**.

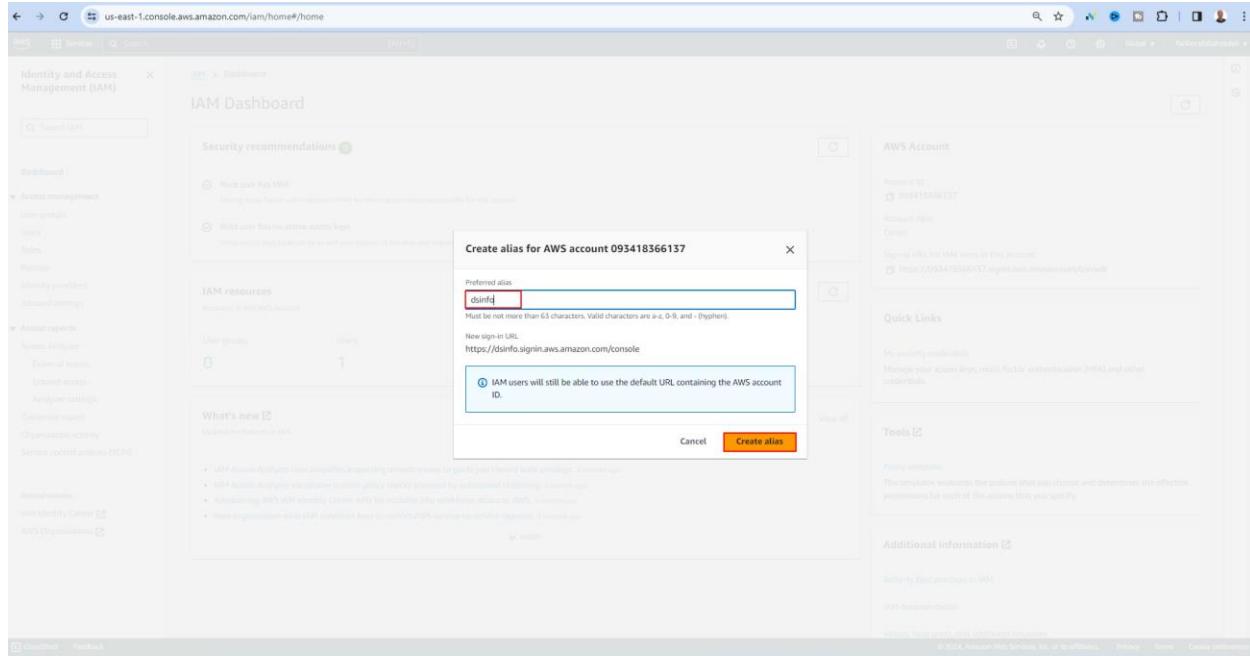
Currently, since the service usage is less than 5 dollars, the alarm status is shown as **OK**.



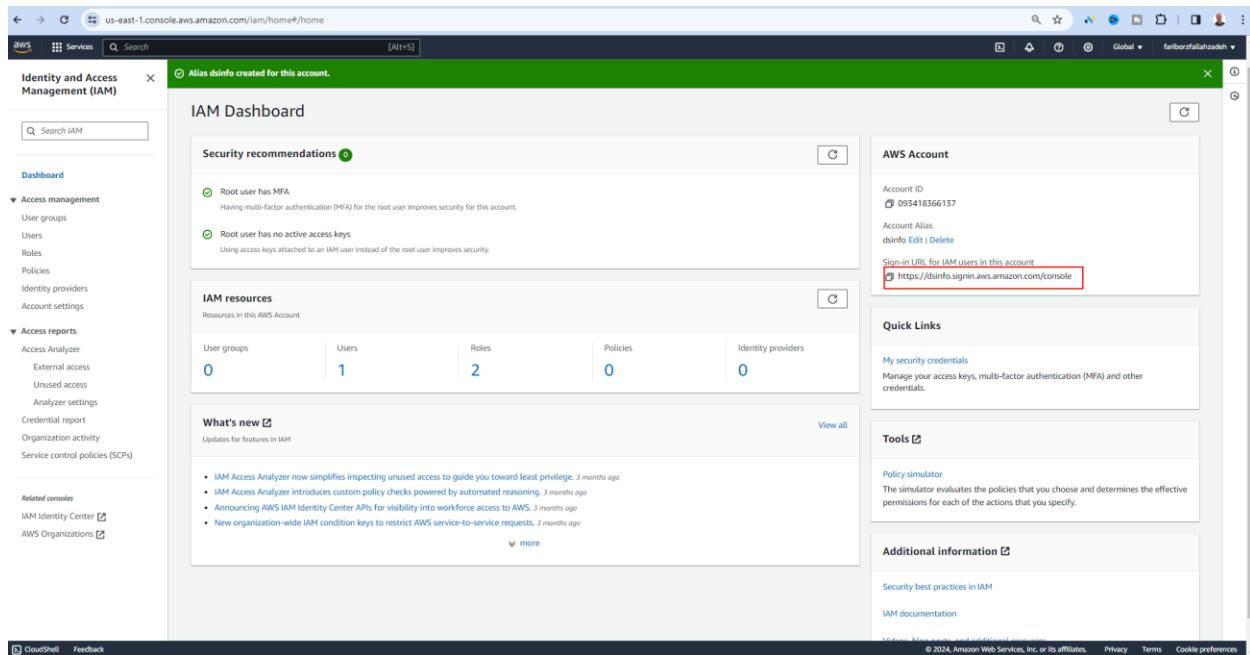
All the AWS setup steps have been completed, and now you can log in to the **AWS Management Console** using the **IAM User** through the link shown in the image. You can also set up an alias for this link.

The screenshot shows the AWS IAM Dashboard. On the left, there's a sidebar with navigation links like 'Identity and Access Management (IAM)', 'Dashboard', 'Access management', 'Access reports', and 'Related consoles'. The main content area has sections for 'Security recommendations' (with two items: 'Root user has MFA' and 'Root user has no active access keys'), 'IAM resources' (showing 0 User groups, 1 User, 2 Roles, 0 Policies, and 0 Identity providers), and 'What's new' (listing recent updates). To the right, there are several panels: 'AWS Account' (Account ID: 093418366137, Account Alias: Create, Sign-in URL: https://093418366137.signin.aws.amazon.com/console), 'Quick Links' (My security credentials, Policy simulator, Additional information), 'Tools' (Policy simulator), and 'Additional information' (Security best practices in IAM, IAM documentation, Videos, blog posts, and additional resources). At the bottom, there are links for CloudShell, Feedback, Privacy, Terms, and Cookie preferences.

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**.

The screenshot shows the AWS Identity and Access Management (IAM) service interface. On the left, there's a navigation sidebar with sections like 'Identity and Access Management (IAM)', 'Access management', 'Access reports', and 'Related consoles'. Under 'Access management', the 'Users' section is selected and highlighted with a red box. The main content area displays a table titled 'Users (1) info' with one row for 'itadmin'. The 'itadmin' row is also highlighted with a red box. The table includes columns for 'User name', 'Path', 'Group', 'Last activity', 'MFA', 'Password age', 'Console last sign-in', 'Access key ID', 'Active key age', and 'Access key last used'. At the top right of the table, there are buttons for 'Create user' and 'Delete'.

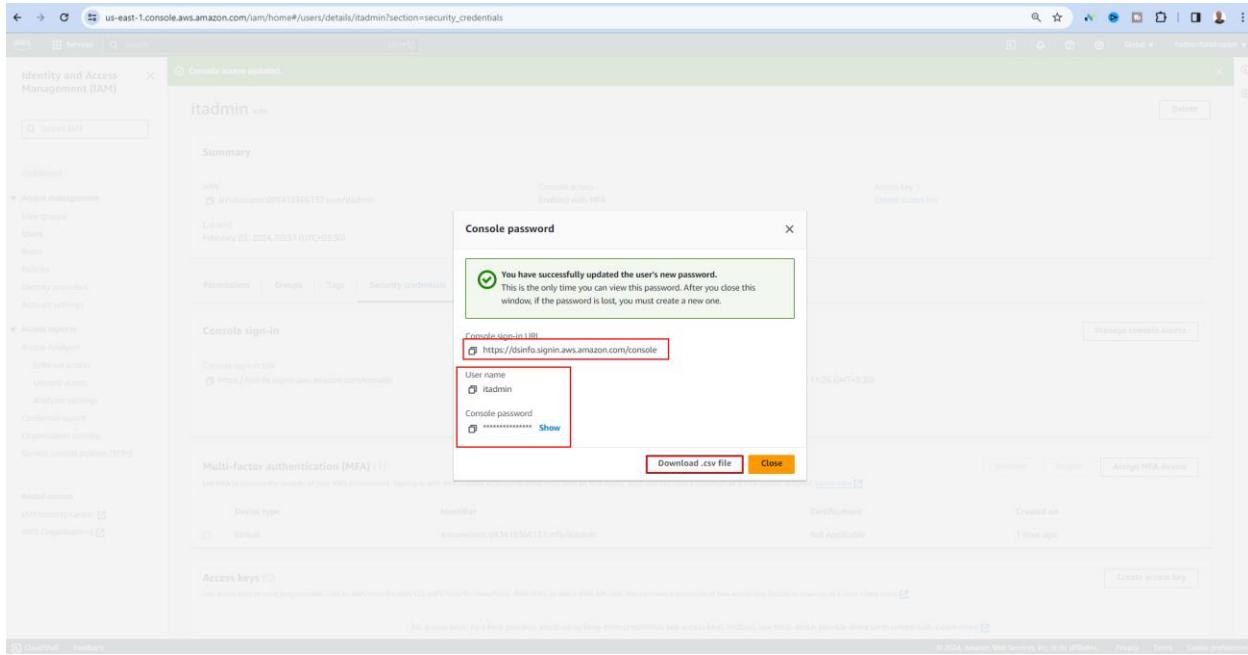
In the **Security Credentials** section, click on the **Manage Console Access** button.

This screenshot shows the detailed view for the 'itadmin' user. The left sidebar remains the same as the previous screenshot. The main area is titled 'itadmin info' and contains a 'Summary' section with details like ARN, creation date, and console access status. Below this is the 'Security credentials' tab, which is highlighted with a red box. This tab contains sections for 'Console sign-in' and 'Multi-factor authentication (MFA)'. In the 'Console sign-in' section, there's a 'Manage console access' button highlighted with a red box. The 'Multi-factor authentication (MFA)' section shows one device type: 'Virtual' with identifier 'arn:aws:iam:093418366137:mfa/itadmin'. At the bottom, there's an 'Access keys' section with a 'Create access key' button highlighted with a red box.

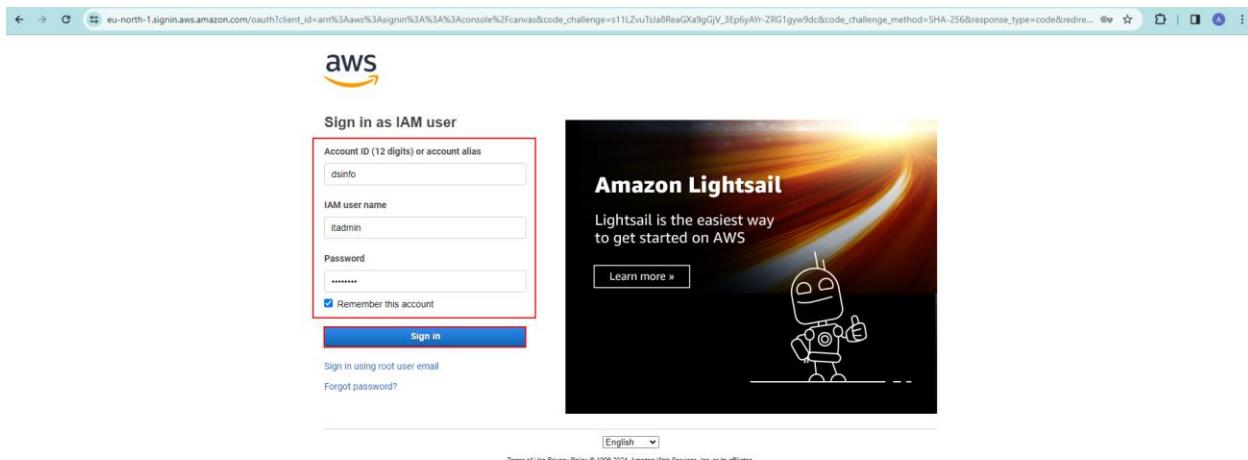
At this stage, select the following options and then click the **Apply** button.

The screenshot shows the AWS IAM console for managing user 'itadmin'. In the 'Console access' section of the 'Manage console access' dialog, the 'Enable' radio button is selected. Under 'Set password', the 'Autogenerated password' radio button is selected and checked. A checkbox labeled 'User must create new password at next sign-in' is also checked. The 'Apply' button is highlighted with a red border.

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

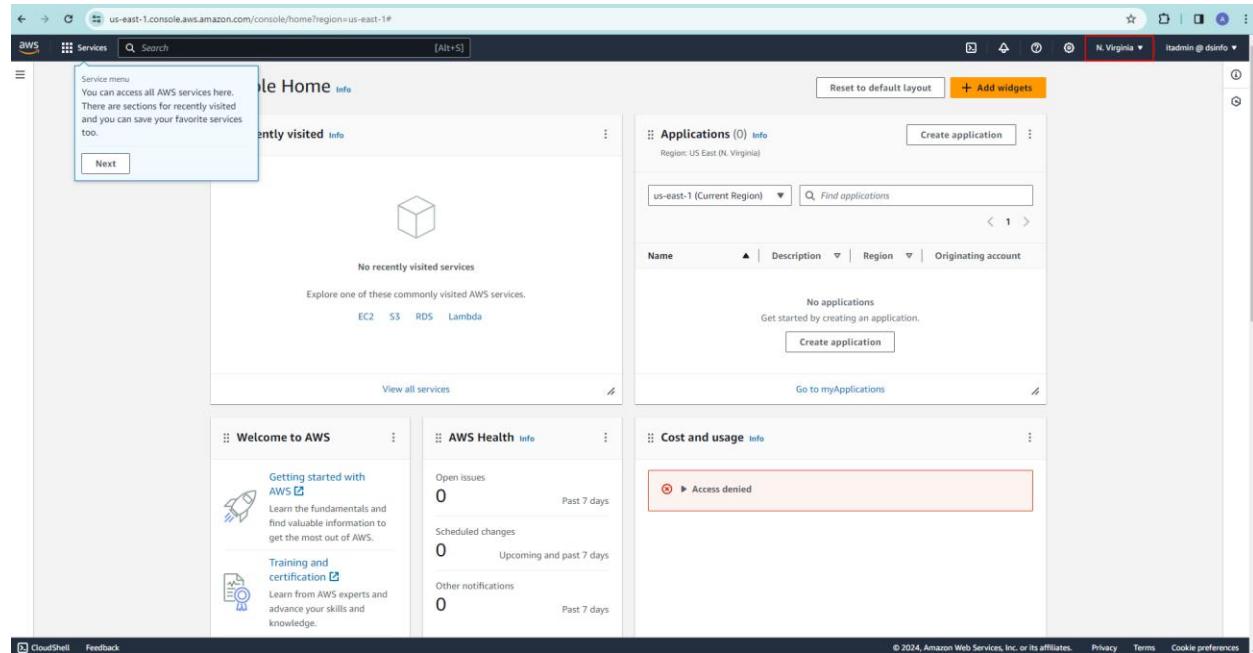


Then, go to the **AWS Management Console** link and sign in using the **IAM User** and **IAM Password** from the CSV file.



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**.

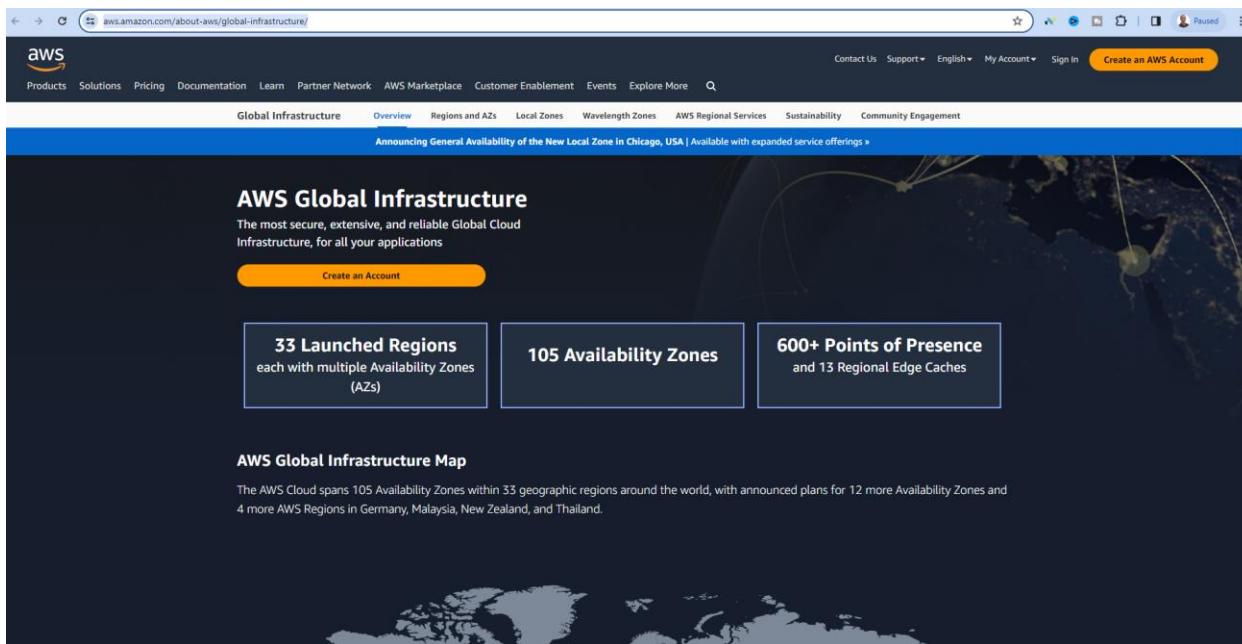
Each Region must have at least two **Availability Zones**.

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.

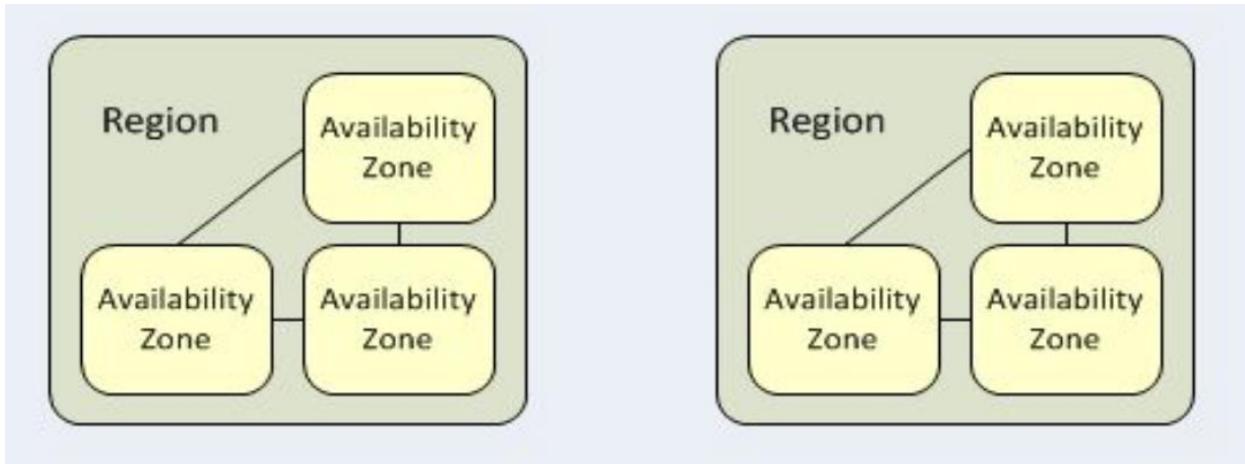
Therefore, if one of your Availability Zones goes down or becomes slow, your web server in the other Availability Zone will remain **up** and operational.

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.

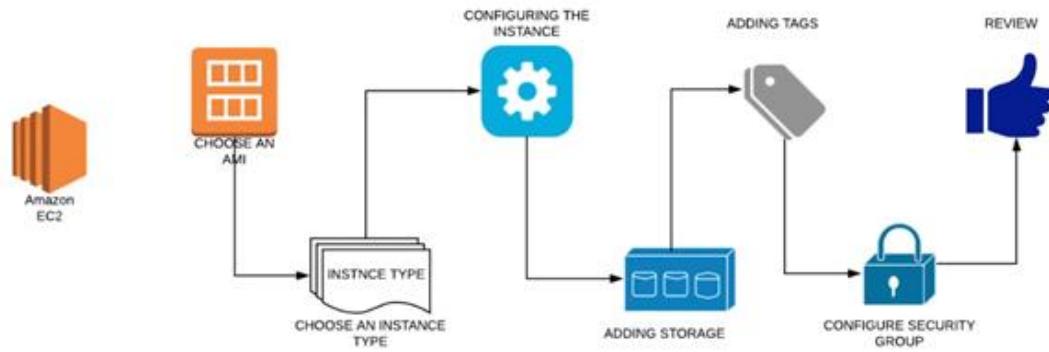
-EC2 Public Key

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**.

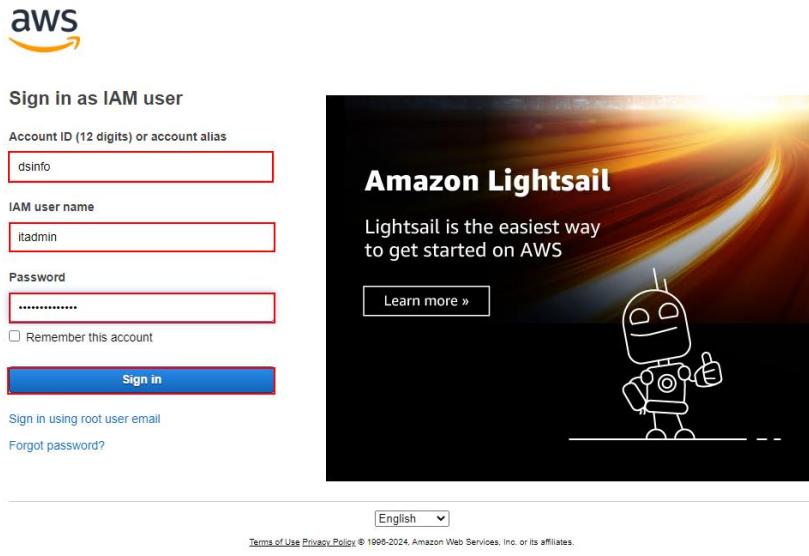
The **EC2 Instance** service refers to a **virtual machine**.

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

The screenshot of the AWS Console Home page shows the 'N. Virginia' region selected in the top right corner. The main dashboard includes sections for 'Recently visited' services (with a placeholder cube icon) and 'Applications' (which is currently empty). Navigation links at the bottom include 'View all services' and 'Go to myApplications'.

Step 2:

Selecting the EC2 Instance Service

The screenshot shows the AWS Management Console search results for the term 'ec2'. The search bar at the top contains 'ec2'. The results are categorized into 'Services' and 'Features'.

Services (13)

- EC2 ☆ Virtual Servers in the Cloud
- EC2 Image Builder ☆ A managed service to automate build, customize and deploy OS images
- Recycle Bin Protect resources from accidental deletion
- Amazon Inspector ☆ Continual vulnerability management at scale

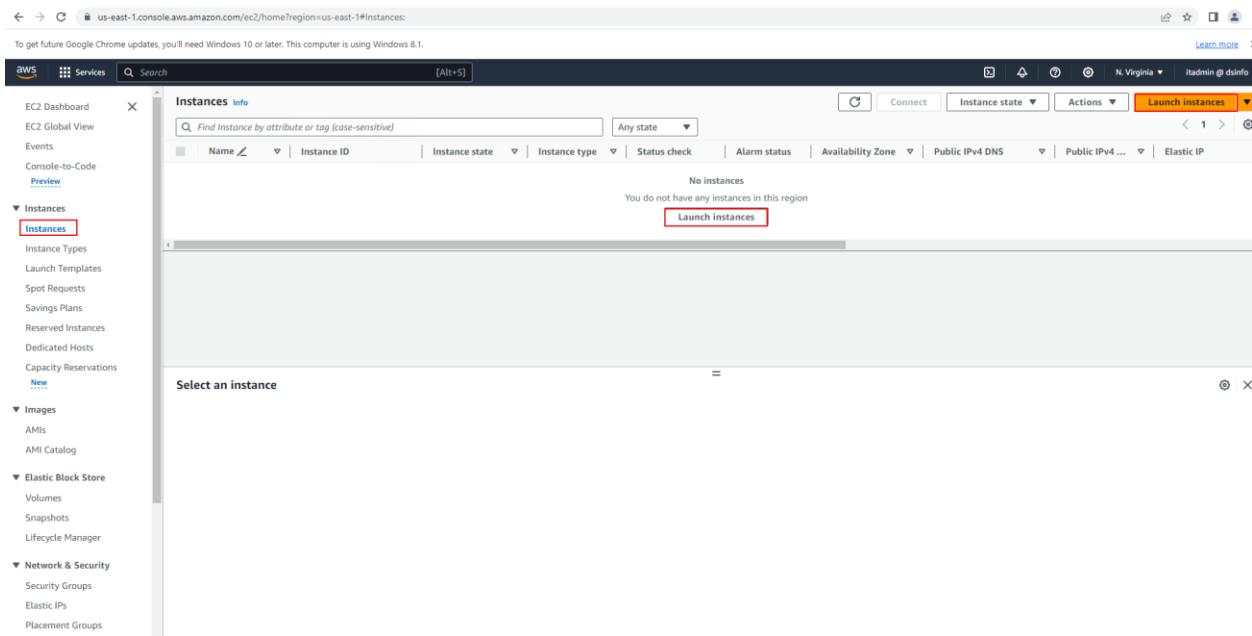
Features (57)

- Dashboard
- EC2 Instances
- AMIs
- Elastic IPs

The result 'EC2 ☆ Virtual Servers in the Cloud' is highlighted with a red box. To the right of the search results, there is a sidebar titled 'Create application' which shows a list of applications with a 'Create application' button.

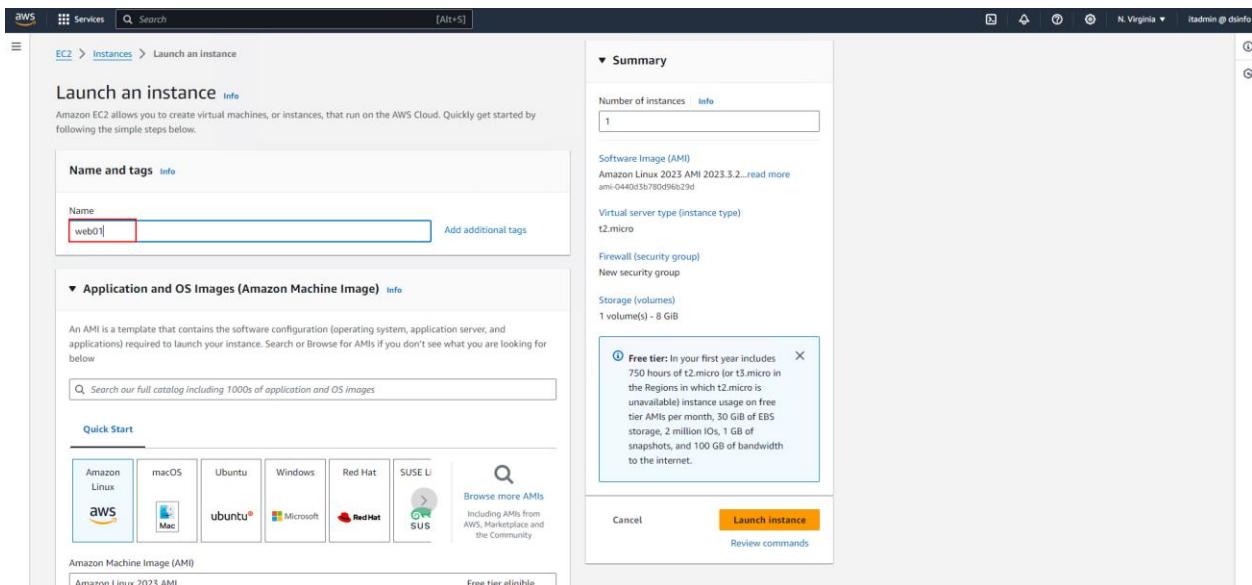
Step 3:

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



Step 4:

Specify a name for the instance.



53

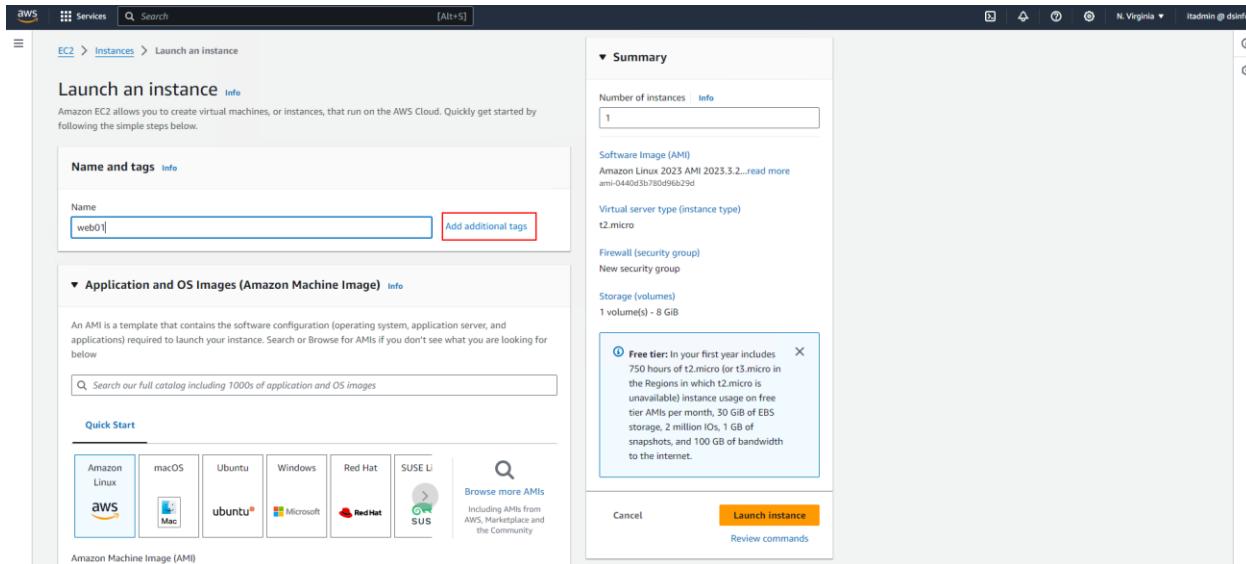
Instructor: Fariborz Fallahzadeh

Email Address:fariborz.fallahzadeh@gmail.com

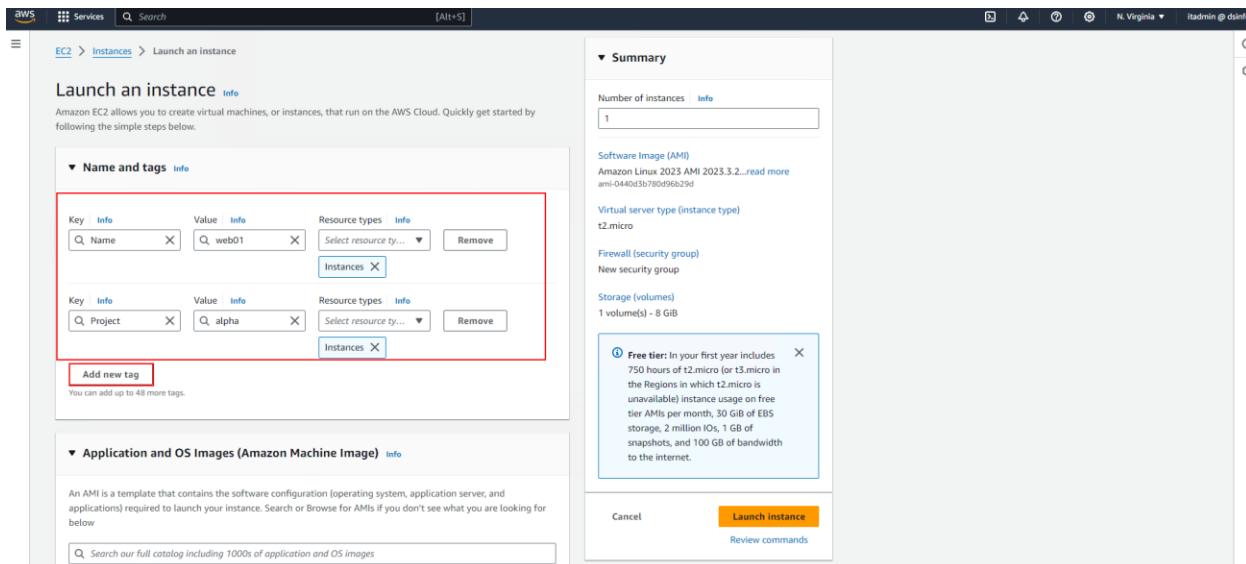
Step 5:

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).

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.

Selected AMI: (ami-0440d3b780d96b29d) (Quickstart AMIs)

Search for an AMI by entering a search term e.g. "Windows"

Quickstart AMIs (47) My AMIs (0) AWS Marketplace AMIs (9614) Community AMIs (500)

Refine results

All products (15 filtered, 47 unfiltered)

Free tier only info

OS category

All Linux/Unix All Windows

Architecture

64-bit (Arm) 52-bit (x86) 64-bit (x86) 64-bit (Mac) 64-bit (Mac-Arm)

aws Amazon Linux 2023 AMI ami-0440d3b780d96b29d (64-bit (x86), uefi-preferred) / ami-0f93c02efdf1974b8b (64-bit (Arm), uefi) Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications. Platform: amazon Root device type: ebs Virtualization: hvm ENA enabled: Yes Select 64-bit (x86), uefi-preferred

aws Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type ami-077613aae34c4478d (64-bit (x86)) / ami-06723030fbcb5c75 (64-bit (Arm)) Amazon Linux 2 comes with five years support. It provides Linux kernel 5.10 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is now under maintenance only mode and has been removed from this wizard. Platform: amazon Root device type: ebs Virtualization: hvm ENA enabled: Yes Select 64-bit (x86) 64-bit (Arm)

Red Hat Red Hat Enterprise Linux 9 (HVM), SSD Volume Type ami-0f630eb57a6ee83 (64-bit (x86)) / ami-0339ee0a14a92573d (64-bit (Arm)) Red Hat Enterprise Linux version 9 (HVM), EBS General Purpose (SSD) Volume Type Platform: rhel Root device type: ebs Virtualization: hvm ENA enabled: Yes Select 64-bit (x86) 64-bit (Arm)

Step 7:

Select the Instance Type.

AM Is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

AMI from catalog Quick Start

Amazon Machine Image (AMI)

al2023-ami-2023.3.20240219.0-kernel-6.1-x86_64 ami-0440d3b780d96b29d

Catalog Published Architecture Virtualization Root device type ENA Enabled

Quickstart AMIs 2024-02-16T21:29:42.00 0Z x86_64 hvm ebs Yes

Instance type

t2.micro Family: t2 1 vCPU 1 GiB Memory Current generation: true On-Demand base pricing: 0.0116 USD per Hour On-Demand SUSE base pricing: 0.0116 USD per Hour On-Demand RHEL base pricing: 0.0116 USD per Hour On-Demand Linux base pricing: 0.0116 USD per Hour

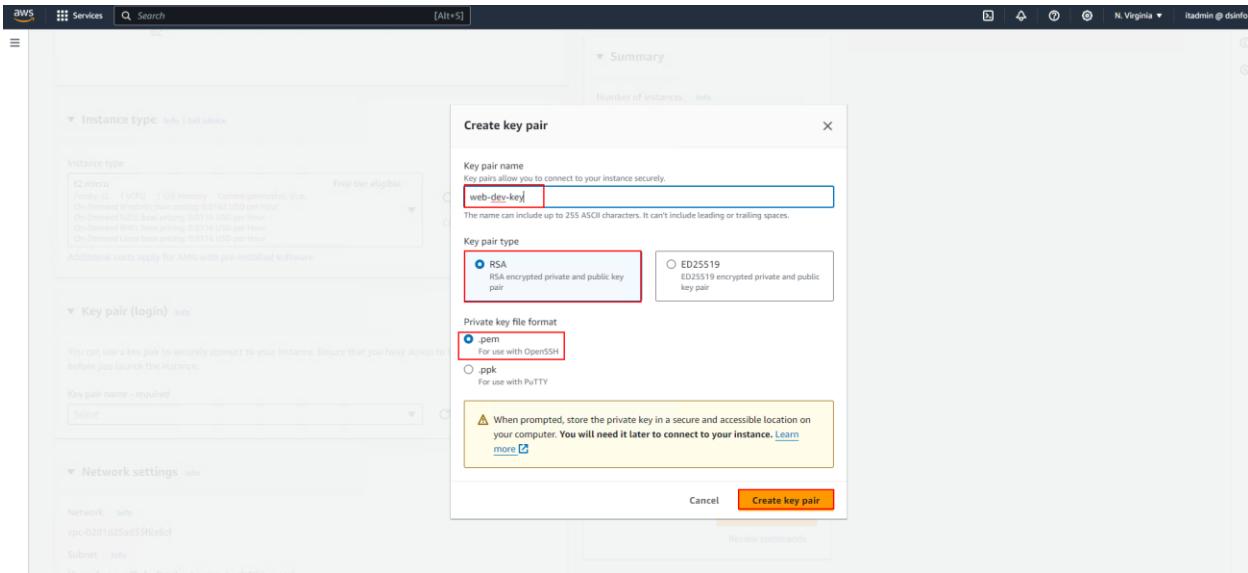
All generations Compare instance types

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GiB of bandwidth to the internet.

Cancel Launch instance Review commands

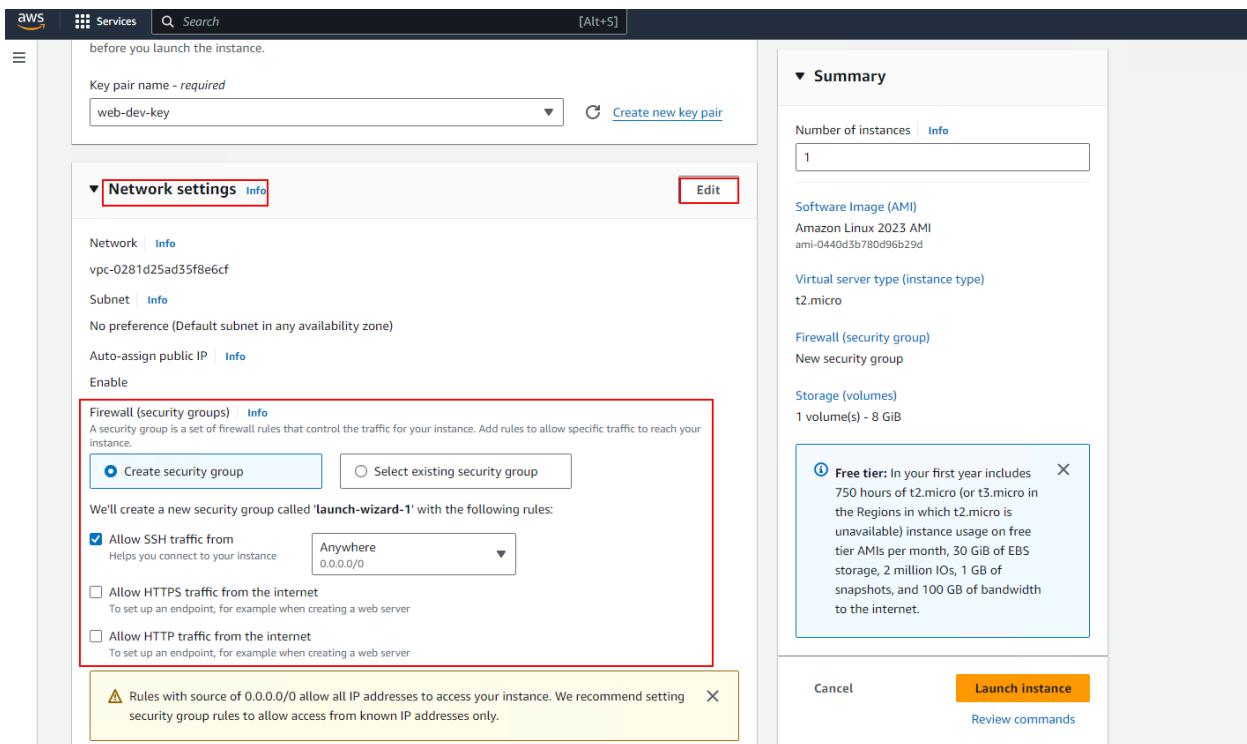
Step 8:

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



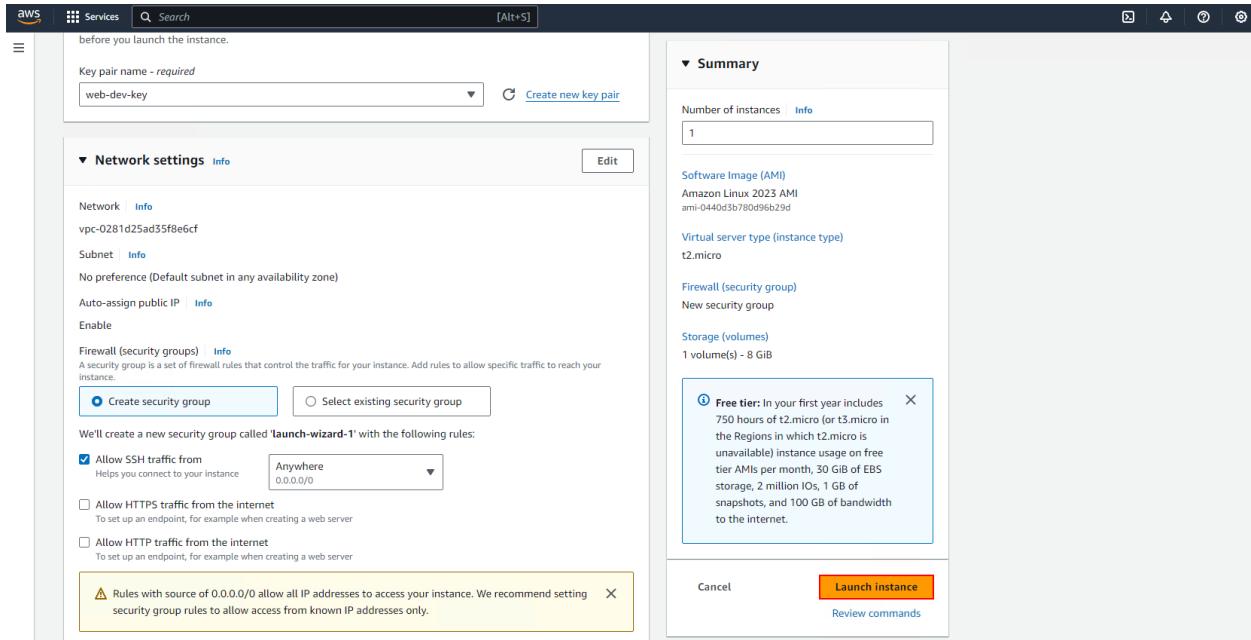
Step 9:

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

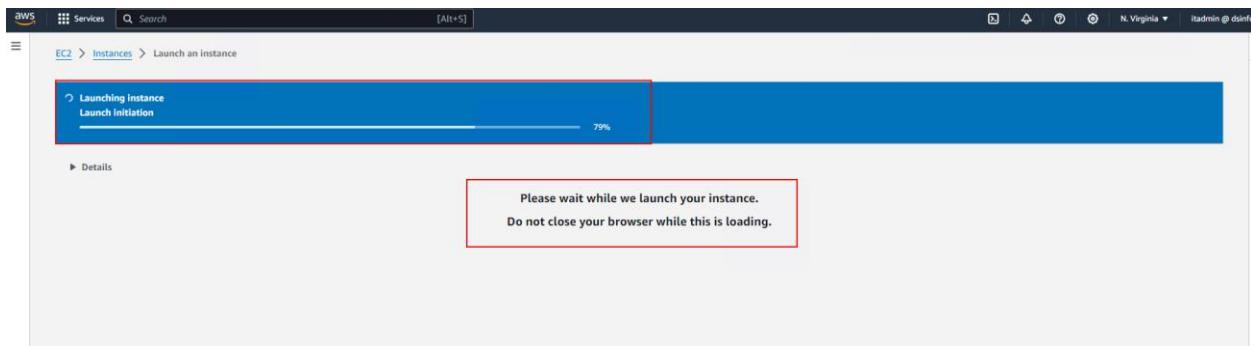


Step 10:

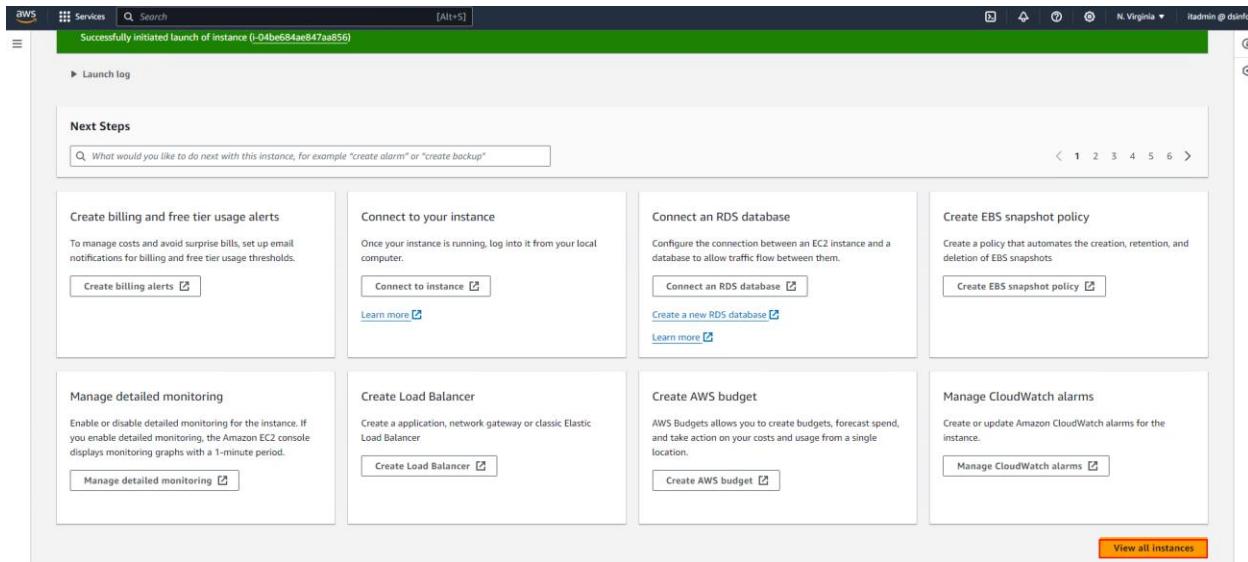
At this stage, click the **Launch Instance** button to create the virtual machine.



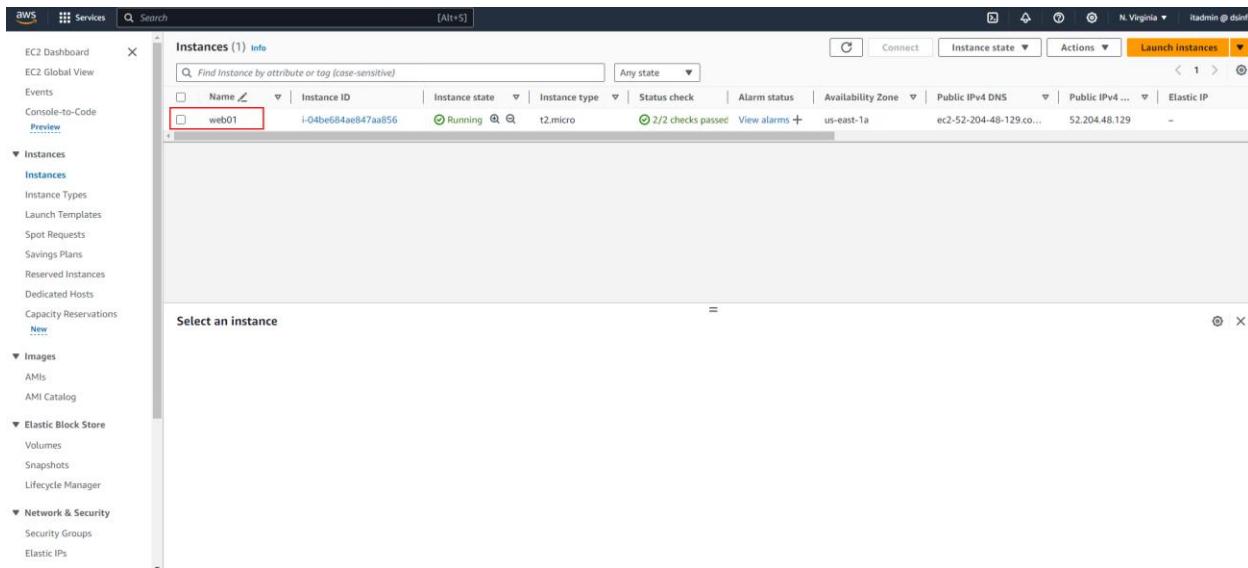
As shown in the image below, the **EC2 instance** is being created.



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**.

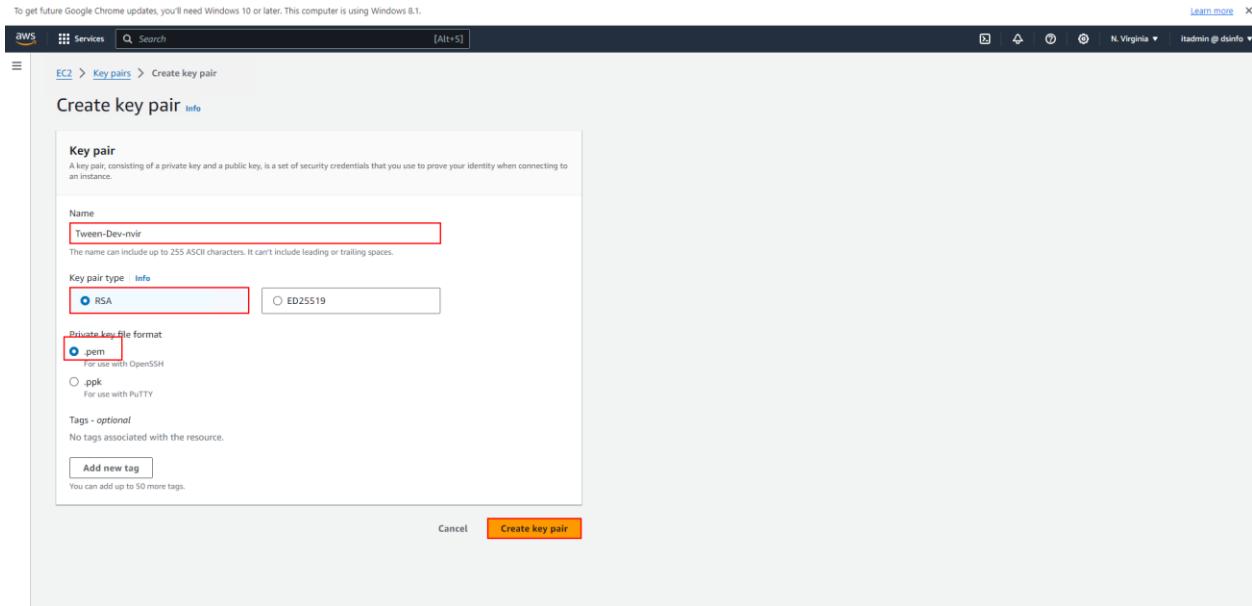
When you launch an instance, you can select and configure all the **requirements** such as the **Security Group**, **Key Pair**, and more.

How to Create a Key Pair

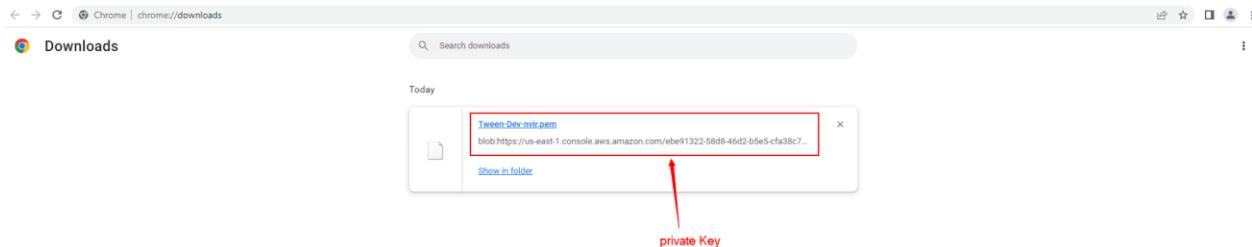
To create a **Key Pair**, go to the **Key Pairs** section and click on the **Create Key Pair** button.

The screenshot shows the AWS Management Console interface. The left sidebar is collapsed. The main navigation bar at the top includes the AWS logo, a search bar, and a 'Services' dropdown. Below the search bar, there's a message: 'To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.' On the right side of the top bar, there are account and region information: 'N. Virginia' and 'itadmin @ dsinfo'. A 'Create key pair' button is highlighted with a red border in the top right corner of the main content area. The main content area has a header 'Key pairs [Info]' with a search bar below it. A table with columns 'Name', 'Type', 'Created', 'Fingerprint', and 'ID' is shown, with the message 'No key pairs to display' above it. The left sidebar contains several service links: Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images (AMIs, AMI Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs - highlighted with a red box), Load Balancing (Load Balancers, Target Groups, Trust Stores), and CloudWatch Metrics. The 'Key Pairs' link under Network & Security is also highlighted with a red box.

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 screenshot shows the AWS EC2 Key Pairs page. On the left, there's a navigation sidebar with links like EC2 Dashboard, Services, Search, and various EC2 management options. The main content area is titled 'Key pairs (1/1) Info'. It displays a table with one row, where the key pair is named 'Twin-Dev-mvir'. A red arrow points from the text 'Public Key' to the 'Name' column of this row. The table also includes columns for Type (rsa), Created (2024/02/26 02:35 GMT-8), Fingerprint (b8:54:e8:8a:e2:00:c1:a1:99:51:3b:e6:2b:79:12:ce:6a:2b:d5:c6), and ID (key-0926f25b0b179c978). There are 'Actions' and 'Create key pair' buttons at the top right of the table.

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.

Name	Security group ID	Security group name	VPC ID	Description	Owner
sg-0ff8e1f663dea4497	sg-0ff8e1f663dea4497	default	vpc-0281d25ad35fBe6ct	default VPC security group	093418366137

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.

EC2 > Security Groups > Create security group

Create security group [Info](#)

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [Info](#)
tween-web-dev-sg

Name cannot be edited after creation.

Description [Info](#)
tween-web-dev-sg

VPC Info
vpc-0281d25ad35f8e6cf

Inbound rules [Info](#)

This security group has no inbound rules.

Add rule

Outbound rules [Info](#)

Type [Info](#) Protocol [Info](#) Port range [Info](#) Destination [Info](#) Description - optional [Info](#)

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

EC2 > Security Groups > Edit security group

Security group name [Info](#)
tween-web-dev-sg

Name cannot be edited after creation.

Description [Info](#)
tween-web-dev-sg

VPC Info
vpc-0281d25ad35f8e6cf

Inbound rules [Info](#)

Type [Info](#) Protocol [Info](#) Port range [Info](#) Source [Info](#) Destination [Info](#) Description - optional [Info](#)

SSH TCP 22 My IP 84.32.10.4/32

Add rule

Outbound rules [Info](#)

Type [Info](#) Protocol [Info](#) Port range [Info](#) Destination [Info](#) Description - optional [Info](#)

All traffic All All Custom 0.0.0.0/0

Add rule

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

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.

Learn more ×

aws Services Search [Alt+S]

Type Info Protocol Info Port range Info Source Info Description - optional Info

SSH TCP 22 My IP 84.32.10.4/32 Delete

Add rule

Outbound rules Info

Type Info Protocol Info Port range Info Destination Info Description - optional Info

All traffic All Custom 0.0.0.0/0 Delete

Add rule

⚠ Rules from source of 0.0.0.0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

Add new tag You can add up to 50 more tags

Cancel Create security group

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

EC2 Dashboard EC2 Global View Events Console-to-Code Preview Instances Instances Instance Types Launch Templates Spot Requests Savings Plans Reserved Instances Dedicated Hosts Capacity Reservations New Images AMIs AMI Catalog Elastic Block Store Volumes Snapshots Lifecycle Manager Network & Security Security Groups Elastic IPs Placement Groups

EC2 > Security Groups > sg-0ea2028daccc1f67c - tween-web-dev-sg

sg-0ea2028daccc1f67c - tween-web-dev-sg

Actions ▾

Details

Security group name tween-web-dev-sg	Security group ID sg-0ea2028daccc1f67c	Description tween-web-dev-sg	VPC ID vpc-0281d25ad55fb66cf
Owner 093418366137	Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry	

Inbound rules | Outbound rules | Tags

Inbound rules (1)

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sgr-0a40ffcb41934f771	IPv4	SSH	TCP	22	84.32.10.4/32	-

Manage tags Edit inbound rules

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

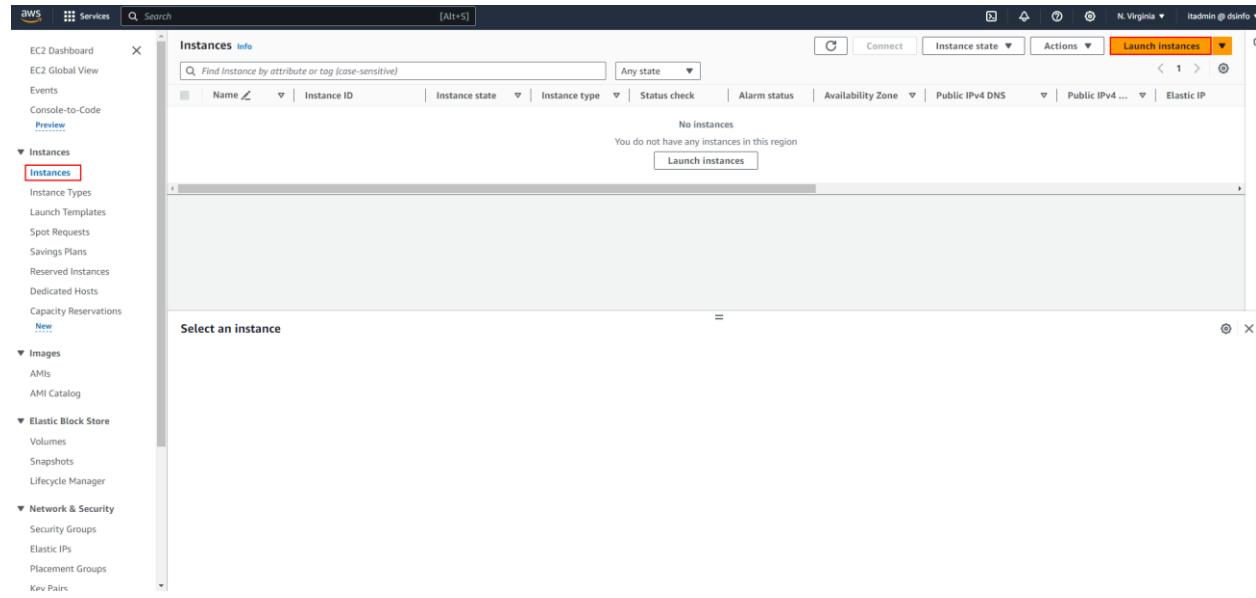
The screenshot shows the AWS EC2 Security Groups interface. On the left, a navigation sidebar lists various services like EC2 Dashboard, Global View, Events, and Network & Security. Under Network & Security, 'Security Groups' is selected. The main content area shows a security group named 'sg-0ea2028dacc1f67c - tween-web-dev-sg'. The 'Details' tab is active, displaying information such as Security group ID (sg-0ea2028dacc1f67c), Description (tween-web-dev-sg), Owner (093418366137), and VPC ID (vpc-0281d25ad358fe6cf). Below the details, there are tabs for 'Inbound rules', 'Outbound rules' (which is currently selected and highlighted with a red border), and 'Tags'. The 'Outbound rules' section shows one rule: 'sgr-025e0fda1da7ba6cd' (IPv4, All traffic, All, 0.0.0.0/0). A search bar and a 'Manage tags' button are also present in this section.

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

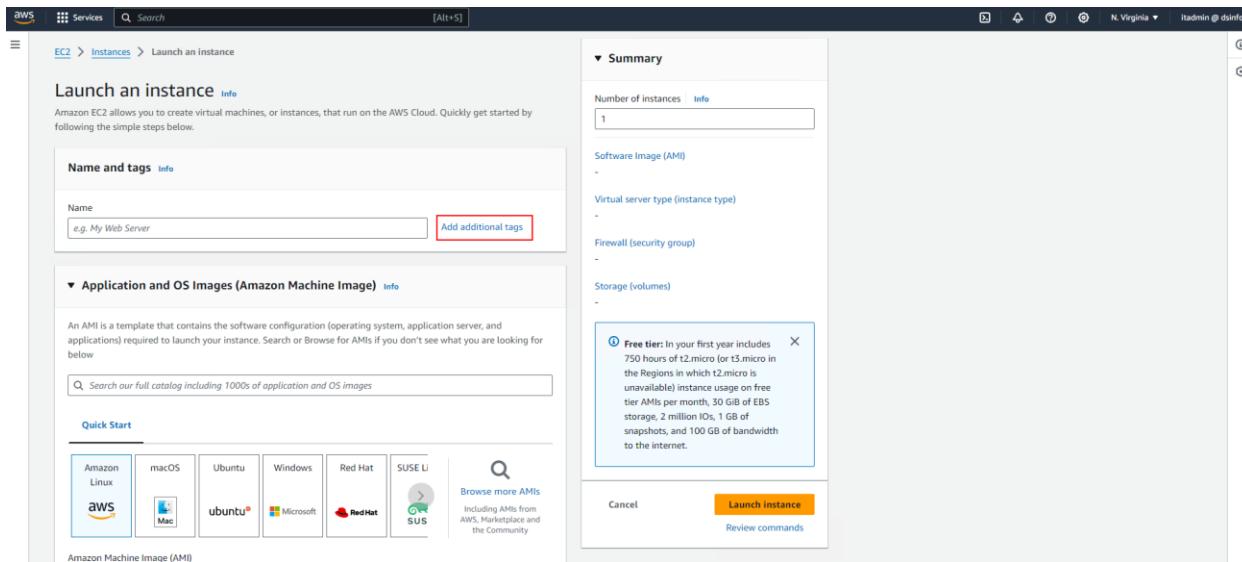
Deploying a Web Server on an EC2 Instance

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

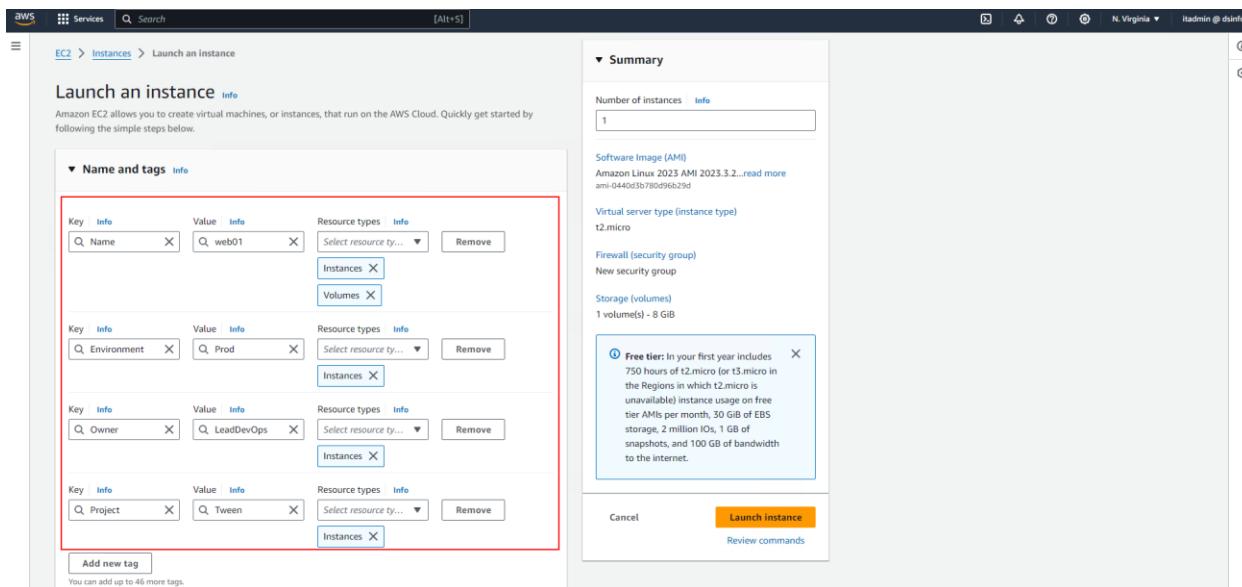
To create an EC2 instance, go to the **Instances** section and click the **Launch Instance** button.



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.



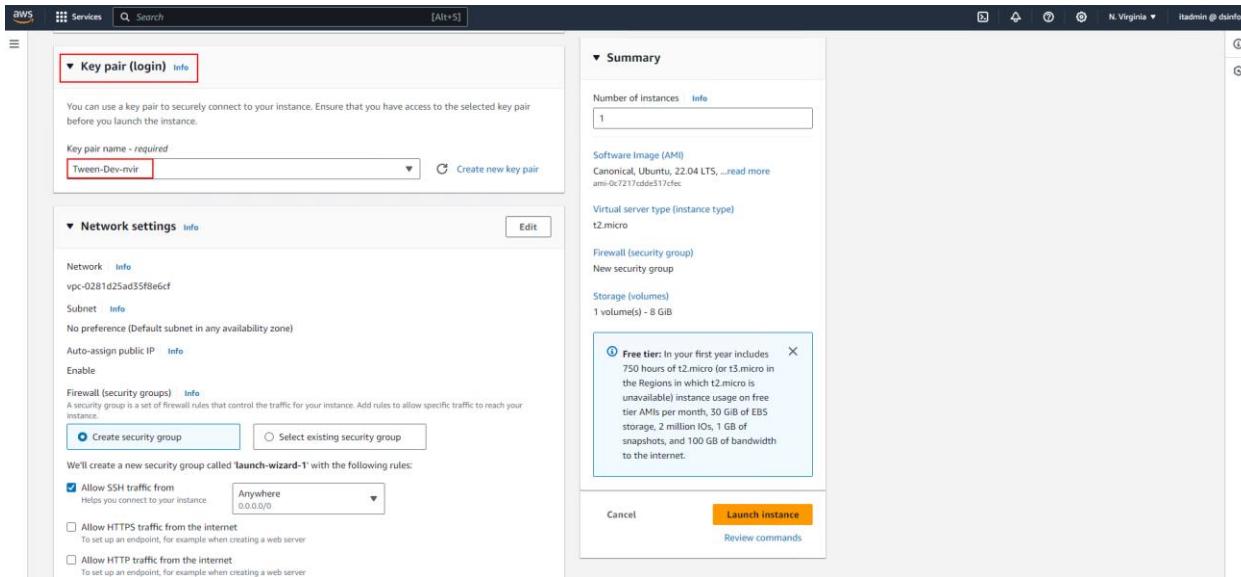
At this stage, you need to select the **AMI type**, and in this scenario, we are choosing **Ubuntu**, which, as you can see, is **Free Tier eligible**.

The screenshot shows the AWS CloudFormation console. On the left, there's a search bar and a list of resources. In the center, under 'Application and OS Images (Amazon Machine Image)', the 'Ubuntu' option is selected and highlighted with a red box. A tooltip for this selection states: 'Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.' On the right, the 'Summary' panel displays the instance configuration: Number of instances: 1, Software Image (AMI): Canonical, Ubuntu, 22.04 LTS, Virtual server type (instance type): t2.micro, Firewall (security group): New security group, Storage (volumes): 1 volume(s) - 8 GiB.

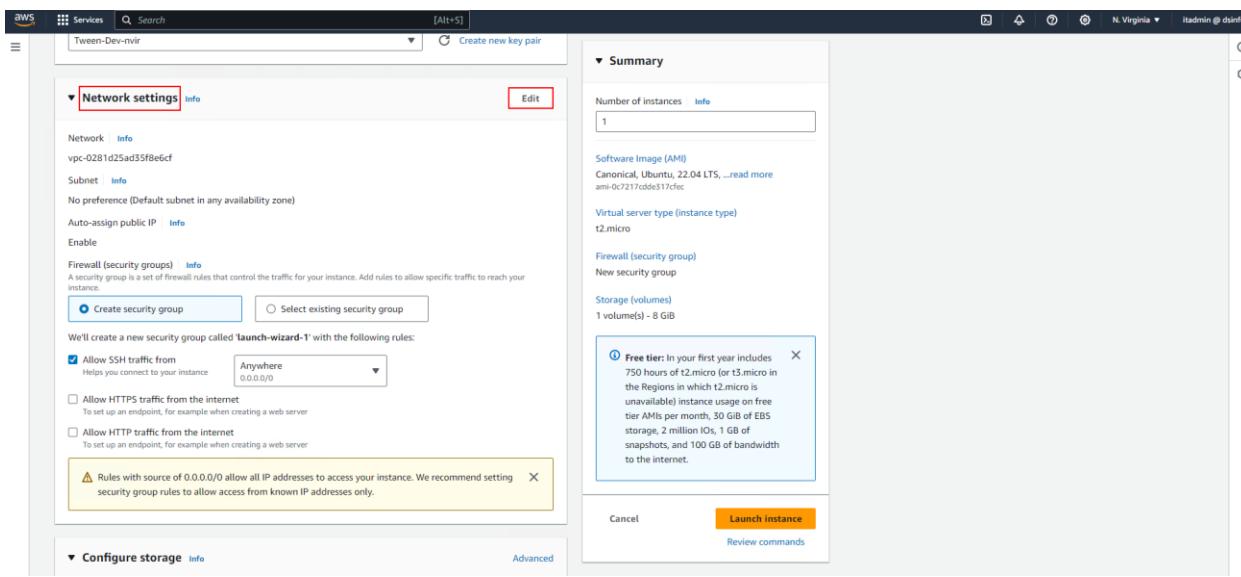
At this stage, you need to specify the **Instance Type**, which defines the hardware specifications of the instance. In this scenario, we are selecting **t2.micro**, which is **Free Tier eligible**.

The screenshot shows the AWS CloudFormation console. Under 'Instance type', the 't2.micro' option is selected and highlighted with a red box. A tooltip for this selection states: 'Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.' On the right, the 'Summary' panel displays the instance configuration: Number of instances: 1, Software Image (AMI): Canonical, Ubuntu, 22.04 LTS, Virtual server type (instance type): t2.micro, Firewall (security group): New security group, Storage (volumes): 1 volume(s) - 8 GiB.

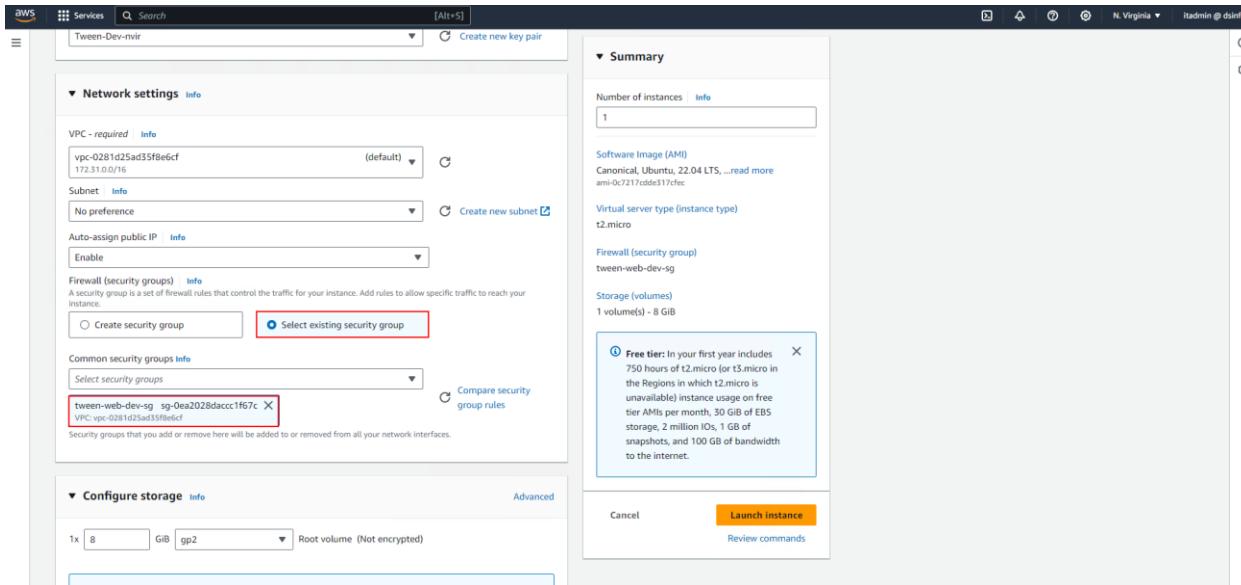
In this section, you need to specify the **Key Pair**, and we will apply the same Key Pair that we created earlier to this instance.



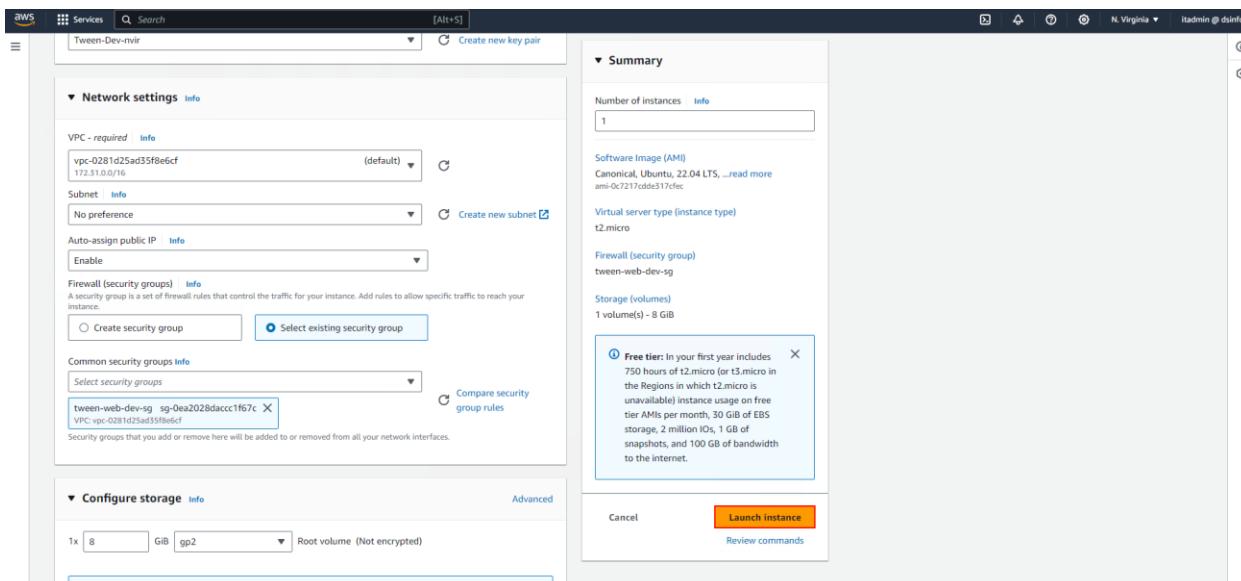
At this stage, to apply the **Security Group**, click on the **Edit** button in the **Network Settings** section.



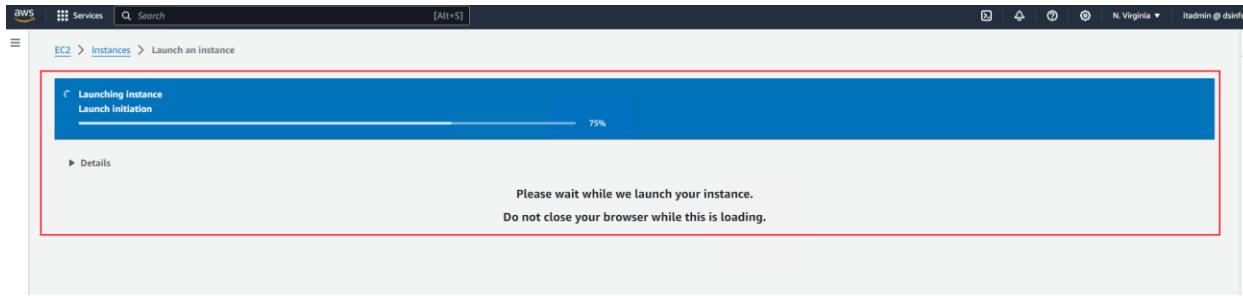
Then, select the **Select Existing Security Group** option, and choose the **Security Group** that we created in the previous steps.



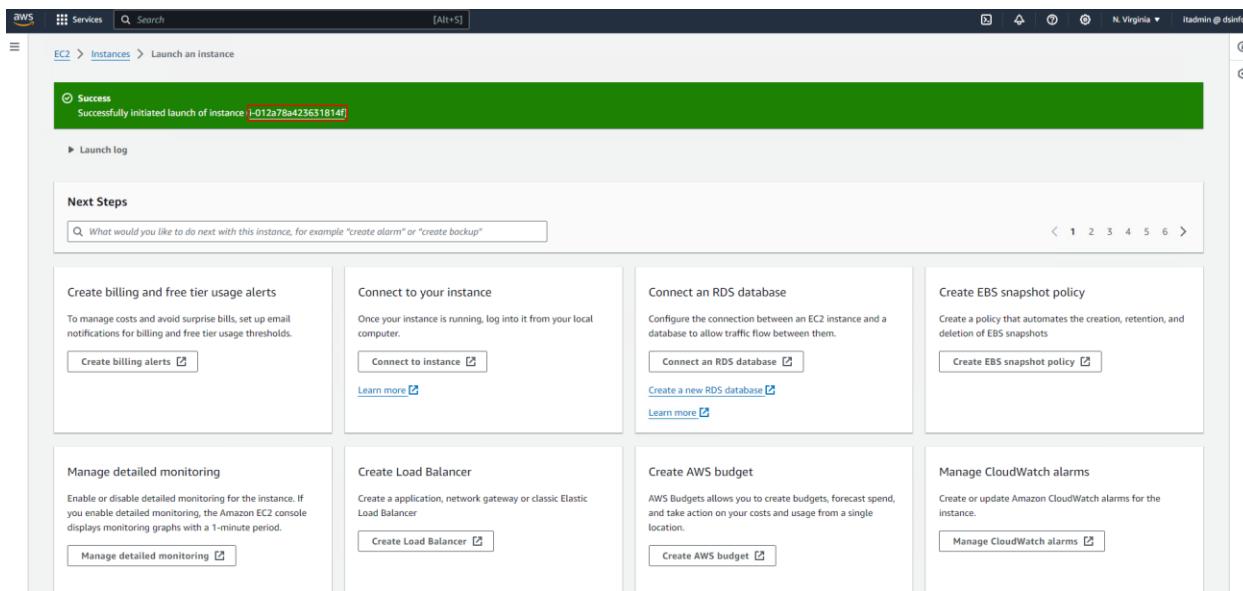
Finally, click on the **Launch Instance** button to create the instance.



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**.



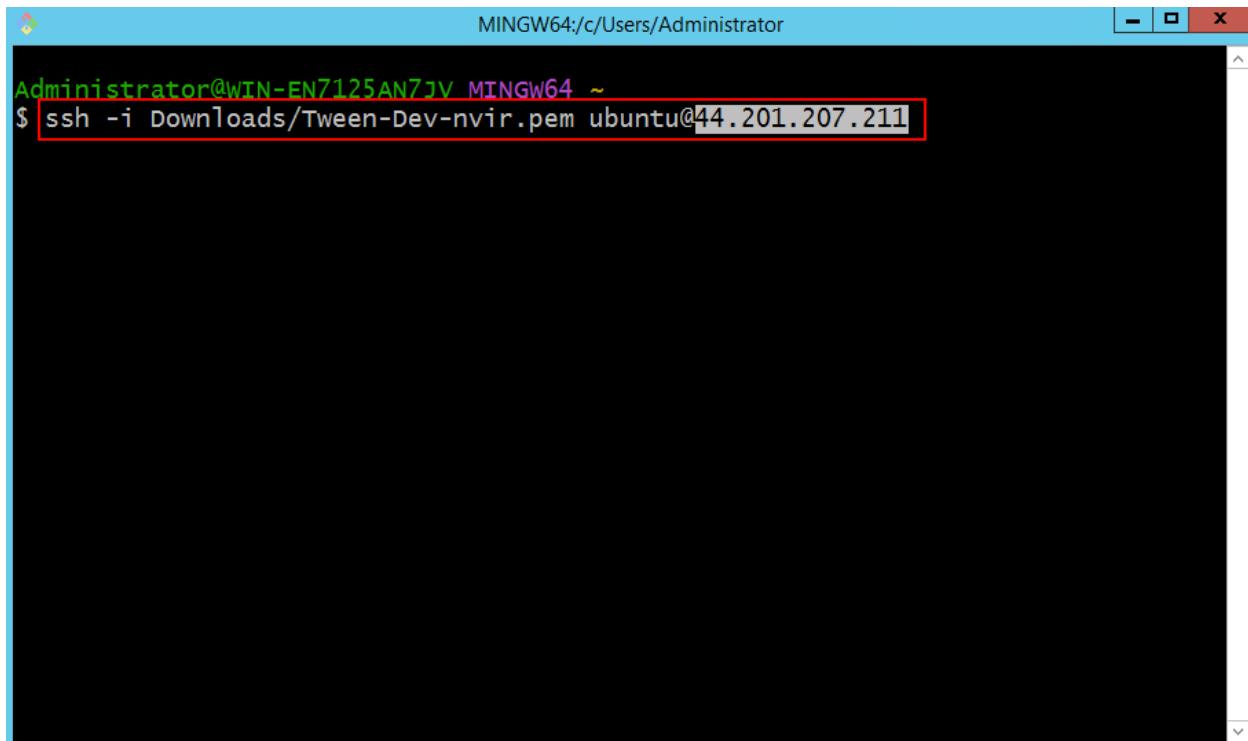
To connect to the instance, click on the **Connect** button.

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with various navigation options like EC2 Global View, Events, Console-to-Code, Instances, Images, Elastic Block Store, Network & Security, and more. The main area displays a table titled 'Instances (1/1) info' with one row for 'web01'. The instance details include its ID (i-012a78a423631814f), state (Running), type (t2.micro), and public IP (44.201.207.211). A red box highlights the 'Connect' button at the top right of the instance card.

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

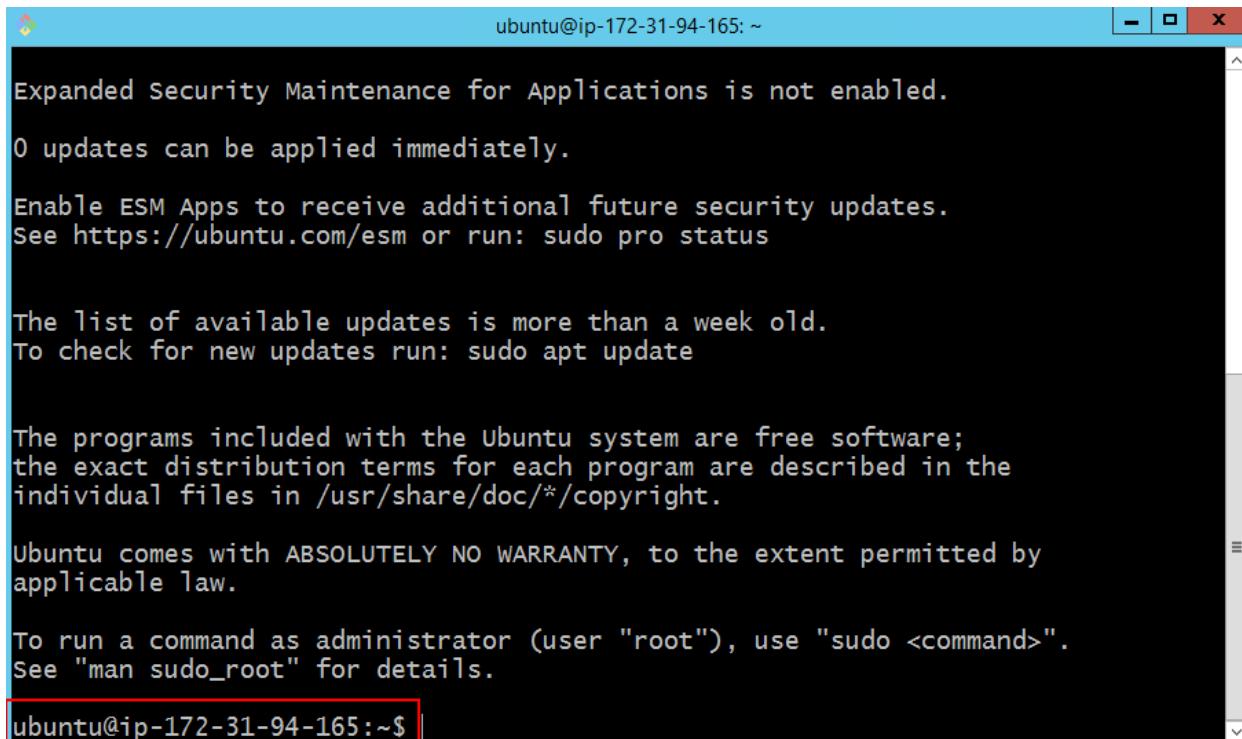
This screenshot shows the 'Connect to instance' dialog for the instance i-012a78a423631814f (web01). It provides instructions for connecting via SSH, mentioning the private key file (Tweed-Dev-nvир.pem) and the public DNS (ec2-44-201-207-211.compute-1.amazonaws.com). A red box highlights the 'SSH client' tab. Two red arrows point from the text 'Private Key' and 'Username' to their respective fields in the 'ssh -i "Tweed-Dev-nvир.pem" ubuntu@ec2-44-201-207-211.compute-1.amazonaws.com' command example.

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.



A screenshot of a terminal window titled "MINGW64:/c/Users/Administrator". The window shows the command line prompt "Administrator@WIN-EN7125AN7JV MINGW64 ~" followed by the command "\$ ssh -i Downloads/Tween-Dev-nvir.pem ubuntu@44.201.207.211". The entire command line is highlighted with a red box.

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



The screenshot shows a terminal window titled "ubuntu@ip-172-31-94-165: ~". The window contains the following text:

```
ubuntu@ip-172-31-94-165: ~
Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

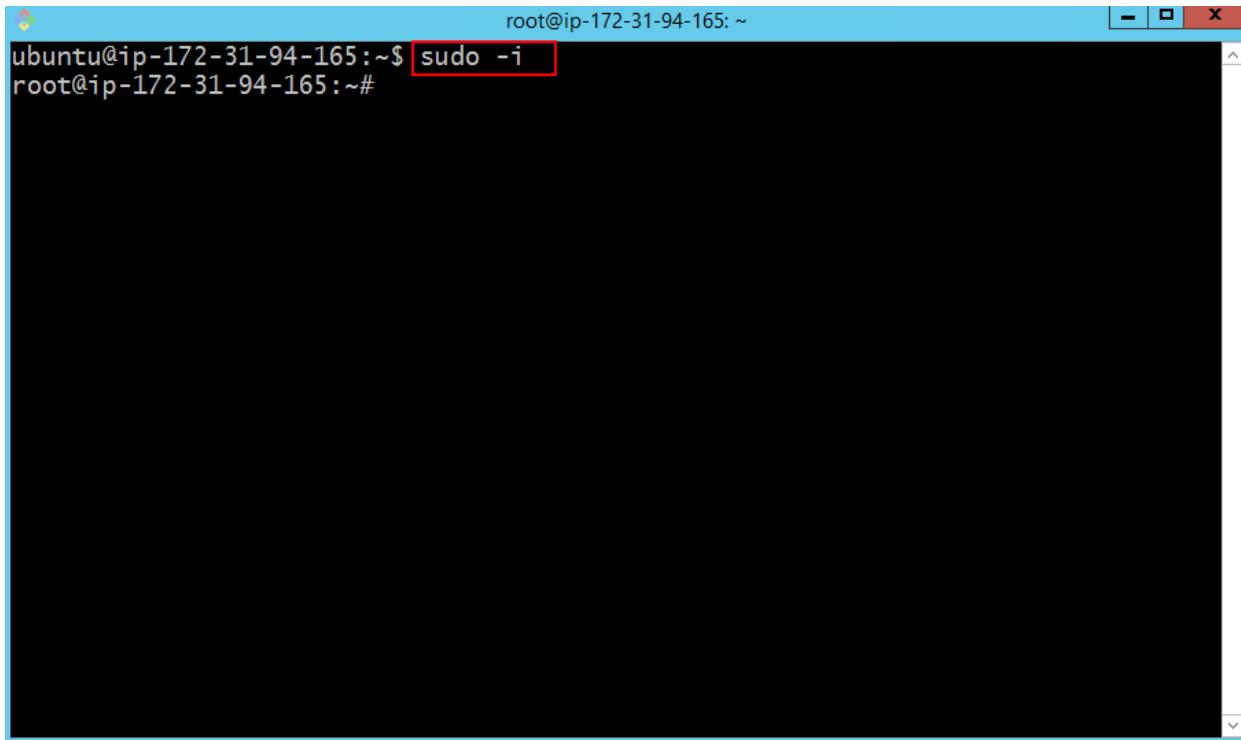
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

The command prompt "ubuntu@ip-172-31-94-165:~\$" is highlighted with a red border.

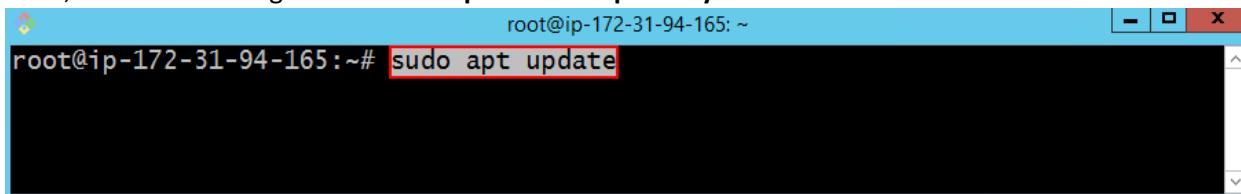
Now, in this section, we will install the **Tween website**, which is one of the templates from **Tooplate**, on this EC2 instance.

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



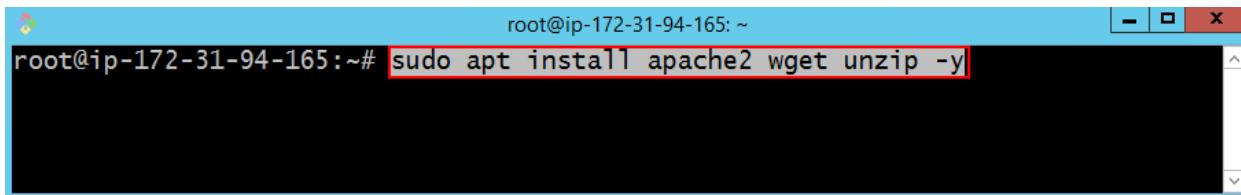
A terminal window titled "root@ip-172-31-94-165: ~". The command "sudo -i" is highlighted with a red box. The prompt "root@ip-172-31-94-165:~#" is visible at the bottom.

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



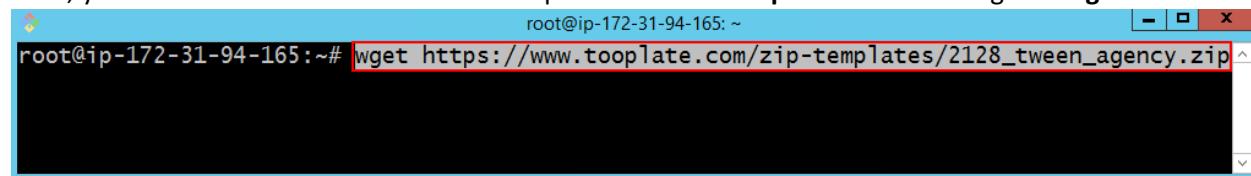
A terminal window titled "root@ip-172-31-94-165: ~". The command "sudo apt update" is highlighted with a red box. The prompt "root@ip-172-31-94-165:~#" is visible at the bottom.

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



A terminal window titled "root@ip-172-31-94-165: ~". The command "sudo apt install apache2 wget unzip -y" is highlighted with a red box. The prompt "root@ip-172-31-94-165:~#" is visible at the bottom.

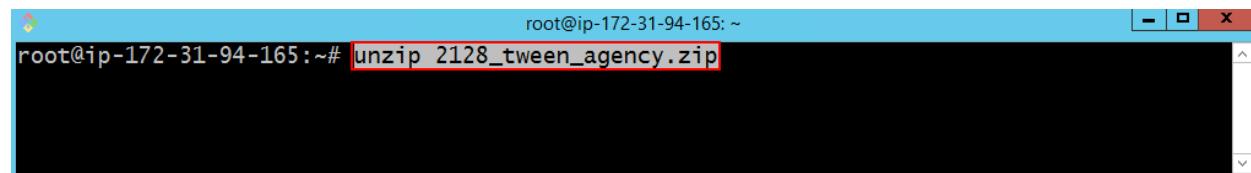
Next, you need to download the **Tween** template from the **Tooplate** website using the **Wget** command.



```
root@ip-172-31-94-165:~# wget https://www.tooplate.com/zip-templates/2128_tween_agency.zip
```

A terminal window titled 'root@ip-172-31-94-165: ~' showing the command 'wget https://www.tooplate.com/zip-templates/2128_tween_agency.zip'. The window has a blue header bar and a black body.

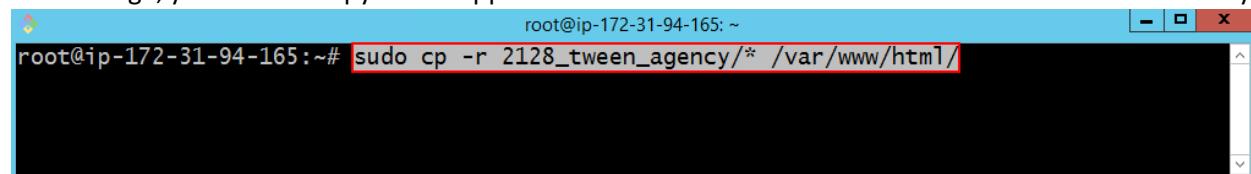
After downloading the **Tween** file, you need to unzip it using the following command



```
root@ip-172-31-94-165:~# unzip 2128_tween_agency.zip
```

A terminal window titled 'root@ip-172-31-94-165: ~' showing the command 'unzip 2128_tween_agency.zip'. The window has a blue header bar and a black body.

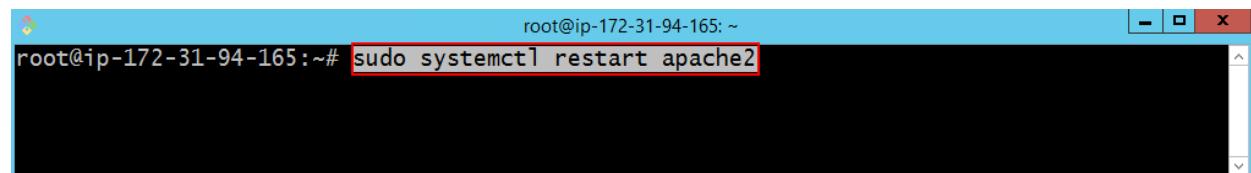
At this stage, you need to copy the unzipped contents of the **Tween** website to the web server's directory.



```
root@ip-172-31-94-165:~# sudo cp -r 2128_tween_agency/* /var/www/html/
```

A terminal window titled 'root@ip-172-31-94-165: ~' showing the command 'sudo cp -r 2128_tween_agency/* /var/www/html/'. The window has a blue header bar and a black body.

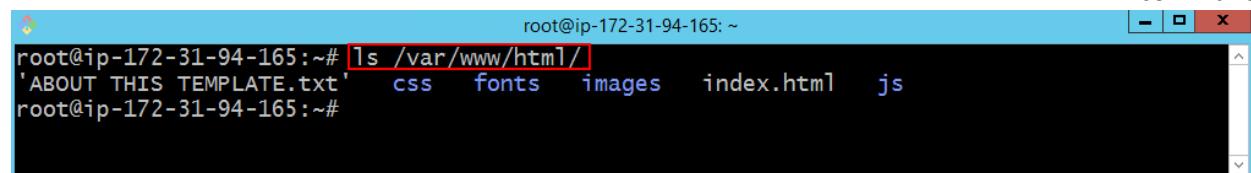
Finally, you need to restart the **Apache** service using the following command



```
root@ip-172-31-94-165:~# sudo systemctl restart apache2
```

A terminal window titled 'root@ip-172-31-94-165: ~' showing the command 'sudo systemctl restart apache2'. The window has a blue header bar and a black body.

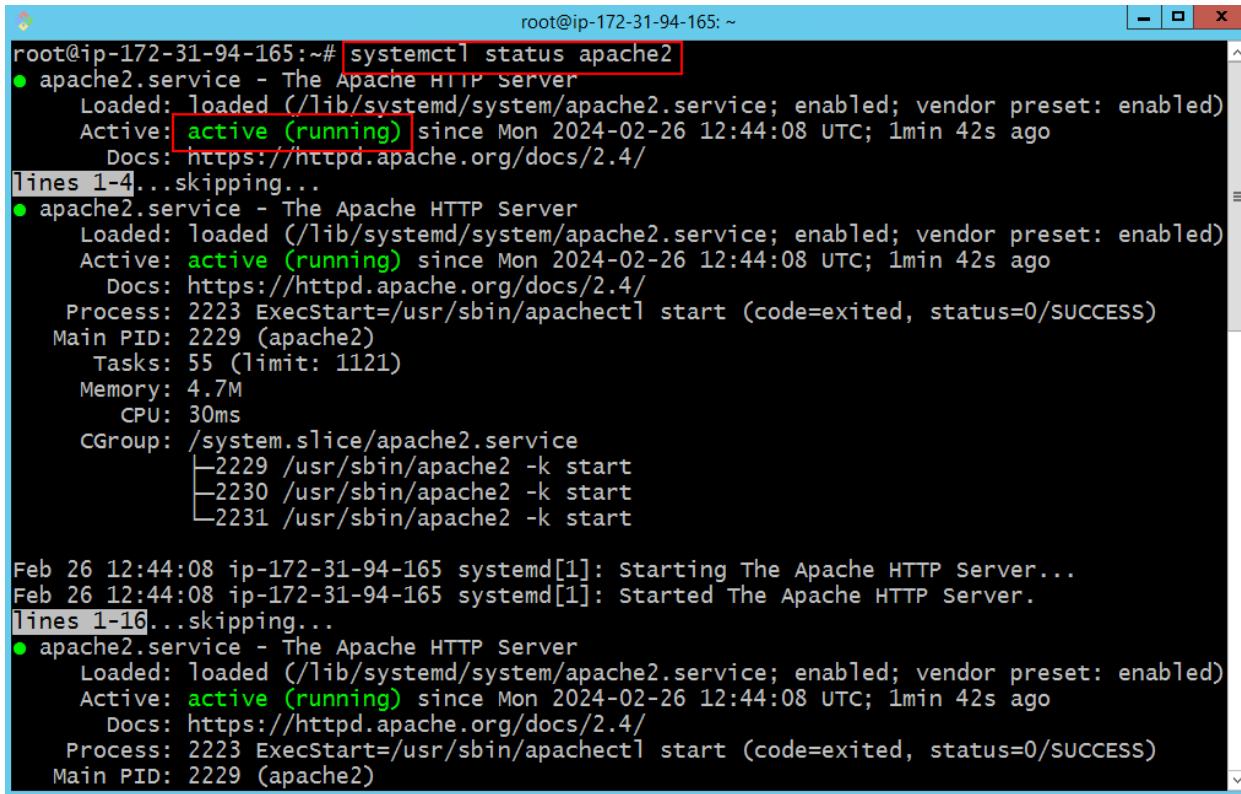
You can check if the **Tween** website files have been copied to the web server directory using the following command



```
root@ip-172-31-94-165:~# ls /var/www/html/
'ABOUT THIS TEMPLATE.txt'  css  fonts  images  index.html  js
root@ip-172-31-94-165:~#
```

A terminal window titled 'root@ip-172-31-94-165: ~' showing the command 'ls /var/www/html/'. The window has a blue header bar and a black body.

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

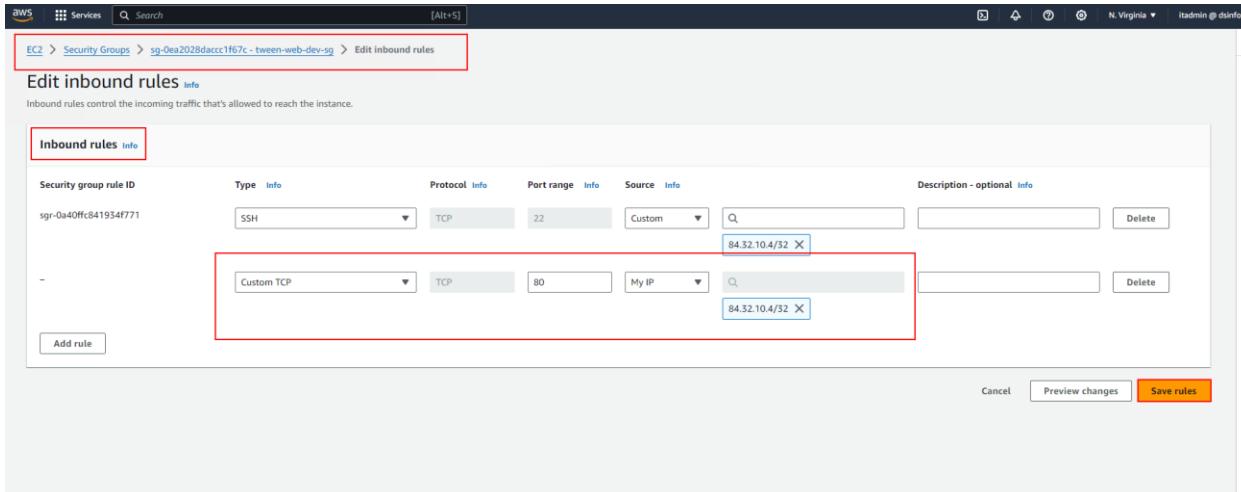


A terminal window titled 'root@ip-172-31-94-165: ~' displaying the output of the 'systemctl status apache2' command. The output shows two active Apache2 services, each with a Main PID of 2229 and a Memory usage of 4.7M. The terminal also shows log entries for the server starting and being active.

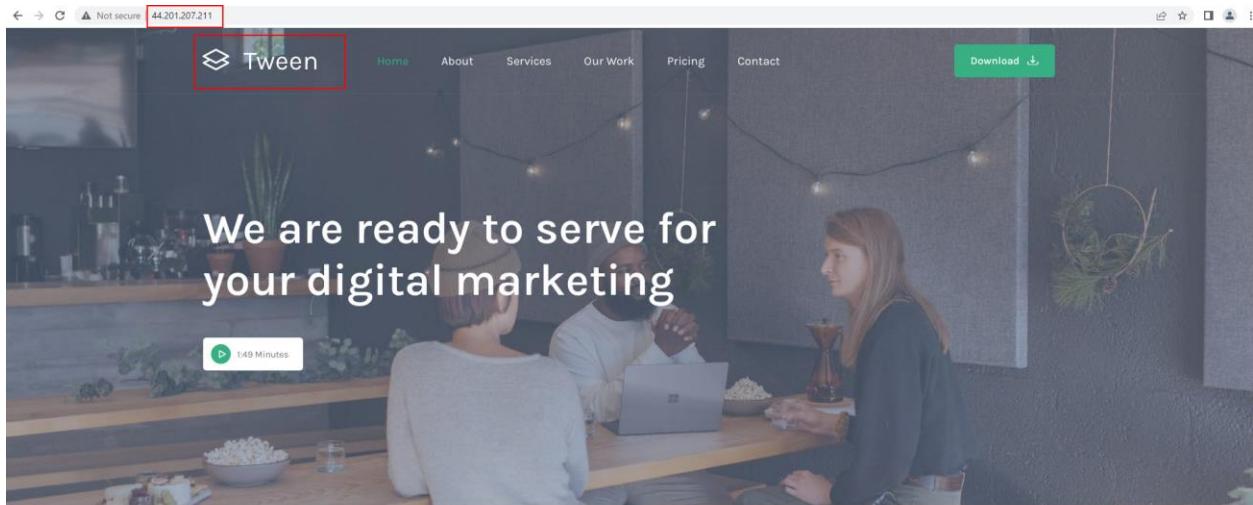
```
root@ip-172-31-94-165:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2024-02-26 12:44:08 UTC; 1min 42s ago
     Docs: https://httpd.apache.org/docs/2.4/
lines 1-4... skipping...
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2024-02-26 12:44:08 UTC; 1min 42s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 2223 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
 Main PID: 2229 (apache2)
    Tasks: 55 (limit: 1121)
   Memory: 4.7M
      CPU: 30ms
     CGroup: /system.slice/apache2.service
             └─2229 /usr/sbin/apache2 -k start
                 ├─2230 /usr/sbin/apache2 -k start
                 ├─2231 /usr/sbin/apache2 -k start

Feb 26 12:44:08 ip-172-31-94-165 systemd[1]: Starting The Apache HTTP Server...
Feb 26 12:44:08 ip-172-31-94-165 systemd[1]: Started The Apache HTTP Server.
lines 1-16... skipping...
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2024-02-26 12:44:08 UTC; 1min 42s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 2223 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
 Main PID: 2229 (apache2)
```

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.



Digital Happiness

[Introduction](#) [Profile](#) [FAQs](#)

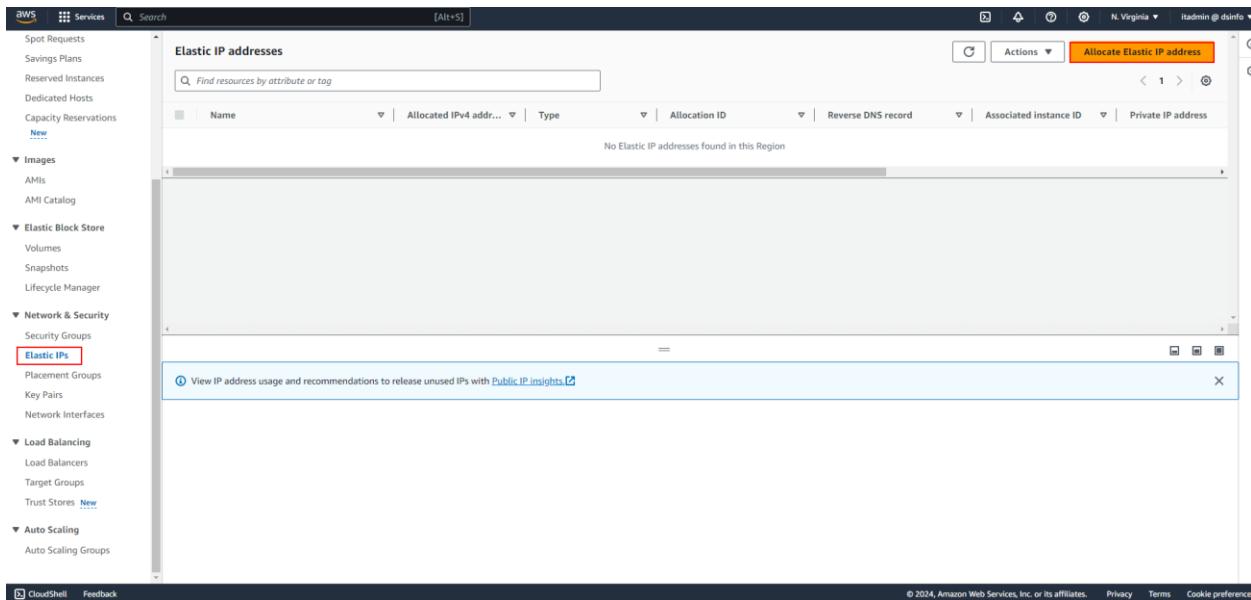
Introduction to Elastic IP

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.

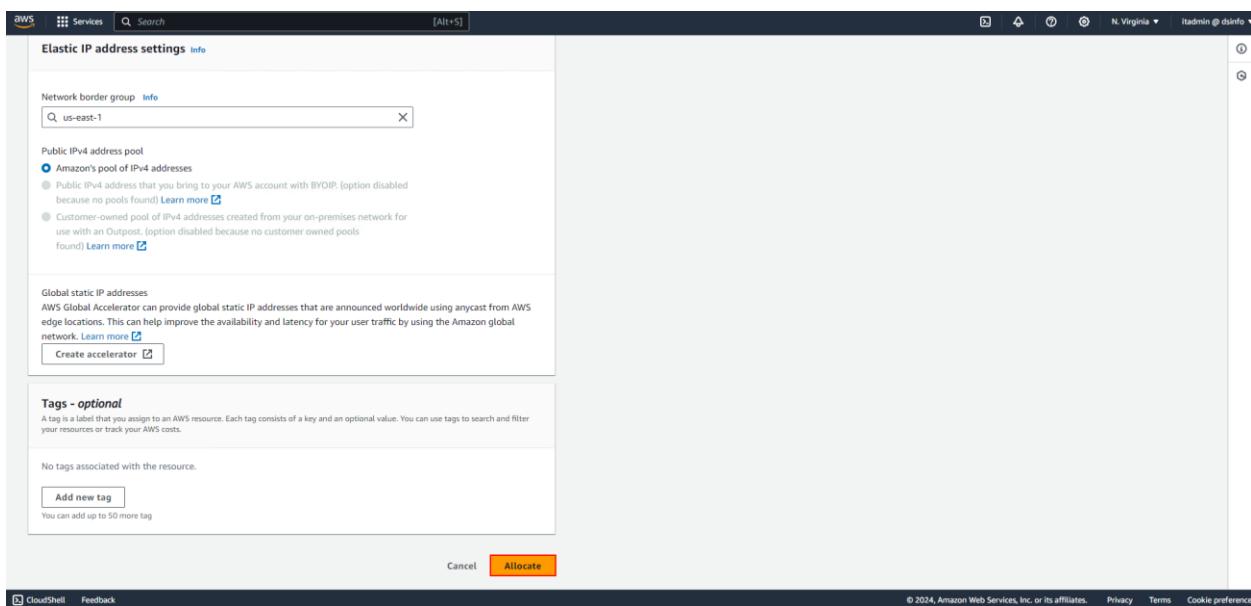
How to Create an Elastic IP

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



The screenshot shows the AWS Management Console interface for managing Elastic IP addresses. The left sidebar navigation includes services like Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security (with 'Elastic IPs' highlighted), Placement Groups, Key Pairs, Network Interfaces, Load Balancing, Target Groups, Trust Stores, Auto Scaling, and Auto Scaling Groups. The main content area displays a table header for 'Elastic IP addresses' with columns: Name, Allocated IPv4 addr..., Type, Allocation ID, Reverse DNS record, Associated instance ID, and Private IP address. A search bar at the top says 'Find resources by attribute or tag'. At the bottom right of the main area is a prominent orange button labeled 'Allocate Elastic IP address'.

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



The screenshot shows the 'Allocate Elastic IP address settings' page. It includes fields for 'Network border group' (set to 'us-east-1'), 'Public IPv4 address pool' (selected 'Amazon's pool of IPv4 addresses'), 'Global static IP addresses' (with a 'Create accelerator' button), and 'Tags - optional' (with an 'Add new tag' button). At the bottom are 'Cancel' and 'Allocate' buttons, with the 'Allocate' button being orange.

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

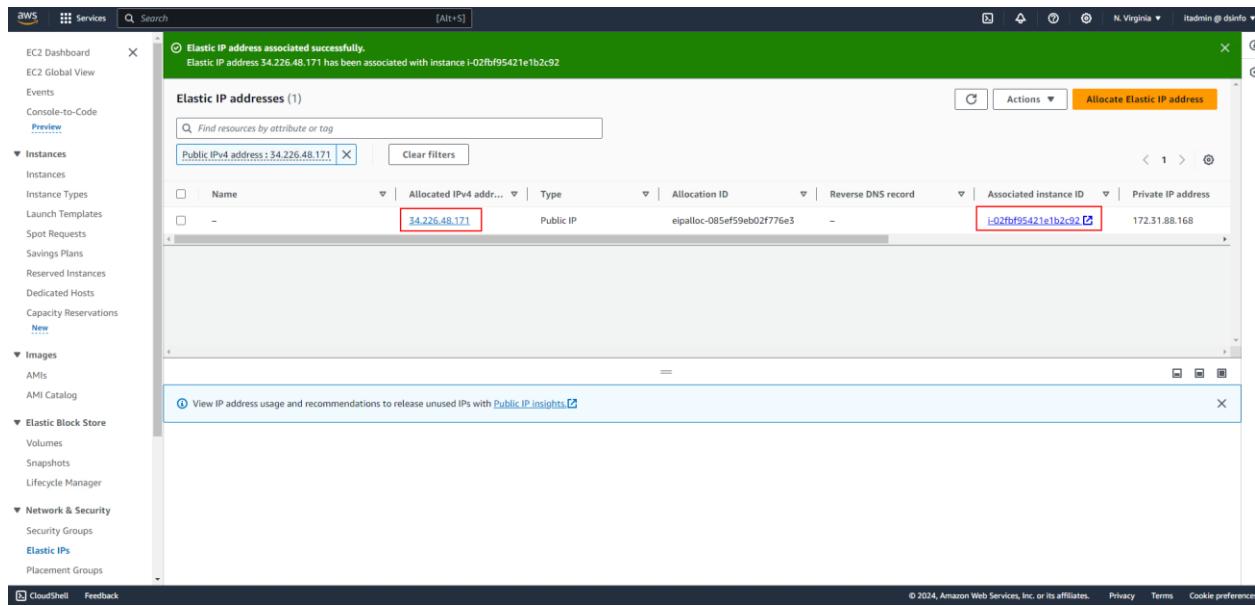
To associate the **Elastic IP** with an instance, go to the **Actions** menu and click on **Associate Elastic IP Address**.

The screenshot shows the AWS EC2 console with the 'Elastic IP addresses' section. A success message at the top says 'Elastic IP address allocated successfully. Elastic IP address 34.226.48.171'. Below it, a table lists one entry: Name (empty), Allocated IPv4 address (34.226.48.171), Type (Public IP), Allocation ID (eipalloc-085ef59eb02f776e3), and Reverse DNS record (empty). To the right of the table is a context menu with several options: 'Associate this Elastic IP address', 'View details', 'Release Elastic IP addresses', 'Associate Elastic IP address' (which is highlighted in red), 'Disassociate Elastic IP address', 'Update reverse DNS', 'Enable transfers', 'Disable transfers', and 'Accept transfers'. At the bottom of the page, there's a summary for the IP address 34.226.48.171, including its allocation ID and type.

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

The screenshot shows the 'Associate Elastic IP address' dialog box. It starts with a header 'Associate Elastic IP address' and a note: 'Choose the instance or network interface to associate to this Elastic IP address (34.226.48.171)'. Below this, a section titled 'Elastic IP address: 34.226.48.171' has a 'Resource type' dropdown where 'Instance' is selected. A warning message states: 'If you associate an Elastic IP address with an instance that already has an Elastic IP address associated, the previously associated Elastic IP address will be disassociated, but the address will still be allocated to your account.' A note below says: 'If no private IP address is specified, the Elastic IP address will be associated with the primary private IP address.' The 'Instance' field contains the ID 'i-02fb95421eb2c9d'. The 'Private IP address' field is empty. There's a checkbox 'Allow this Elastic IP address to be reassociated'. At the bottom are 'Cancel' and 'Associate' buttons.

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



The screenshot shows the AWS Elastic IP Addresses page. At the top, a green banner displays the message: "Elastic IP address associated successfully. Elastic IP address 34.226.48.171 has been associated with instance i-02fbf95421e1b2c92". The main table lists one item:

Name	Allocated IPv4 address	Type	Allocation ID	Reverse DNS record	Associated instance ID	Private IP address
-	34.226.48.171	Public IP	eipalloc-085ef59eb02f776e3	-	i-02fbf95421e1b2c92	172.31.88.168

Below the table, there is a note: "View IP address usage and recommendations to release unused IPs with Public IP insights." The left sidebar shows the navigation menu for EC2 services.

Note:

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

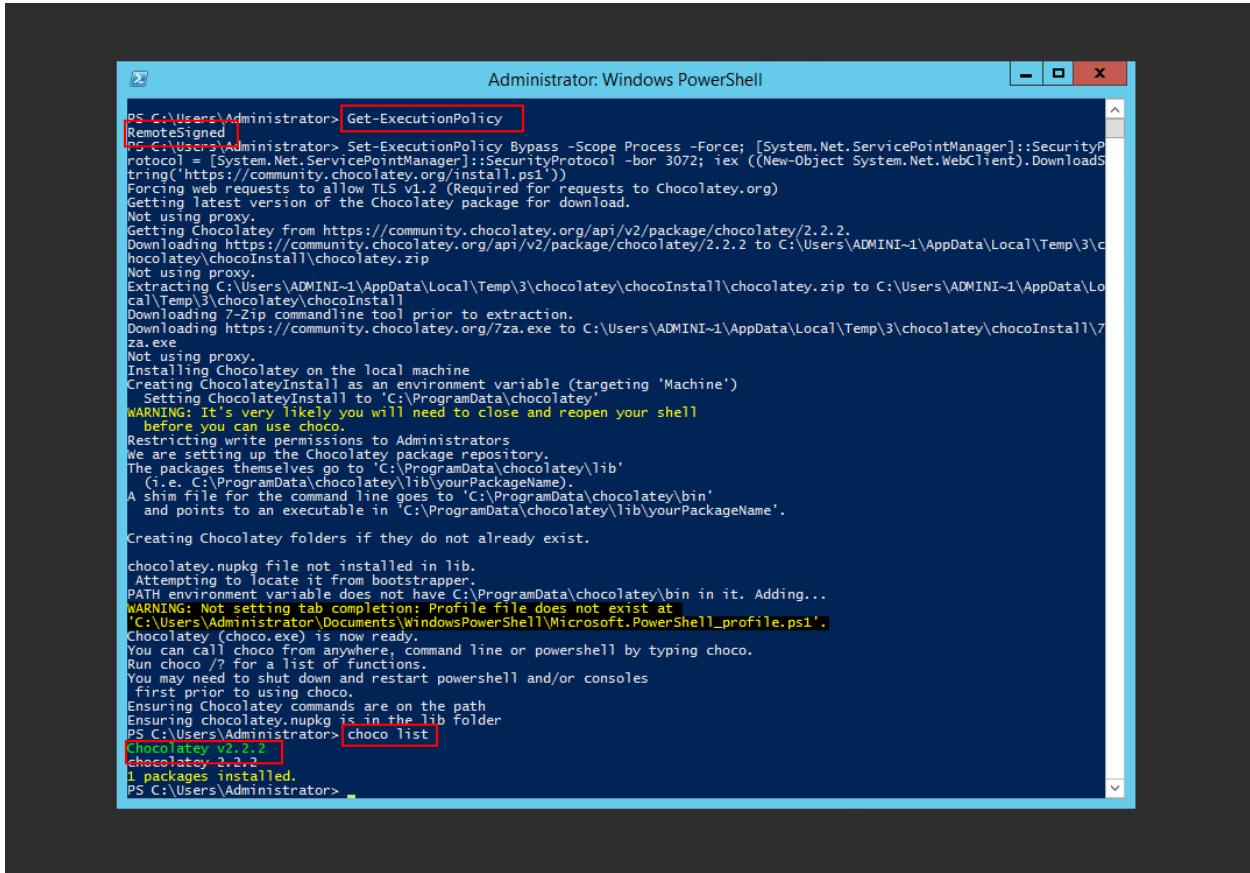
Introduction to AWS CLI

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'))
```

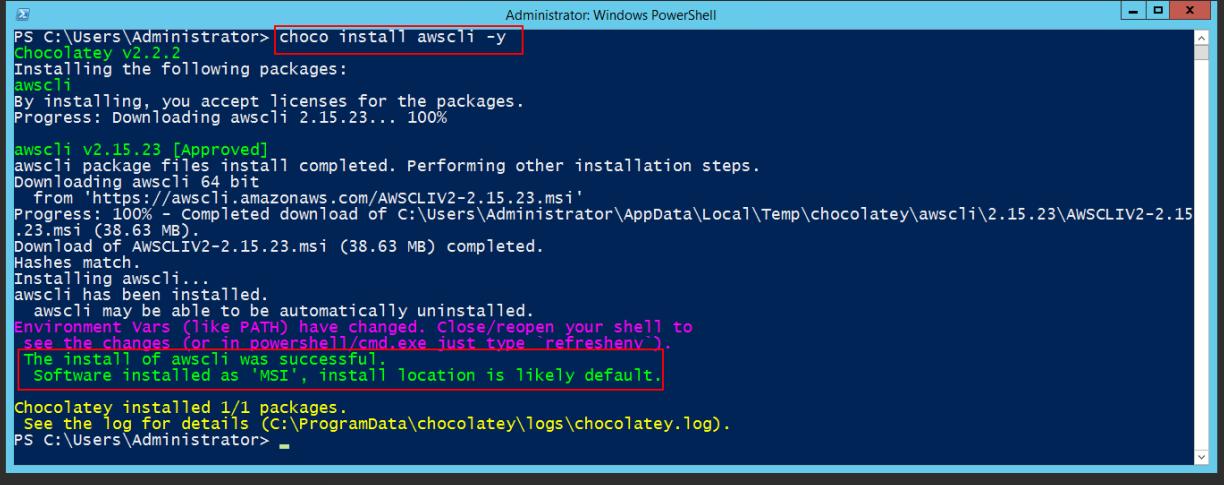


The screenshot shows an Administrator Windows PowerShell window. The command `Get-ExecutionPolicy` is run, showing it is set to `RemoteSigned`. The command `Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))` is then run to bypass the execution policy. The output shows the download and extraction of Chocolatey, including the creation of environment variables and the `choco` command. Finally, the command `choco list` is run, showing one package installed: Chocolatey v2.2.2.

```
Administrator: Windows PowerShell
PS C:\Users\Administrator> Get-ExecutionPolicy
RemoteSigned
PS C:\Users\Administrator> 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'))
Forcing web requests to allow TLS v1.2 (Required for requests to Chocolatey.org)
Getting latest version of the Chocolatey package for download.
Not using proxy.
Getting Chocolatey from https://community.chocolatey.org/api/v2/package/chocolatey/2.2.2.
Downloading https://community.chocolatey.org/api/v2/package/chocolatey/2.2.2 to C:\Users\ADMINI~1\AppData\Local\Temp\3\chocolatey\chocoInstall\chocolatey.zip
Not using proxy.
Extracting C:\Users\ADMINI~1\AppData\Local\Temp\3\chocolatey\chocoInstall\chocolatey.zip to C:\Users\ADMINI~1\AppData\Local\Temp\3\chocoInstall
Downloading 7-Zip commandline tool prior to extraction.
Downloading https://community.chocolatey.org/7za.exe to C:\Users\ADMINI~1\AppData\Local\Temp\3\chocolatey\chocoInstall\7za.exe
Not using proxy.
Installing Chocolatey on the local machine
Creating ChocolateyInstall as an environment variable (targeting 'Machine')
Setting ChocolateyInstall to 'C:\ProgramData\chocolatey'
WARNING: It's very likely you will need to close and reopen your shell
before you can use choco.
Restricting write permissions to Administrators
We are setting up the Chocolatey package repository.
The packages themselves go to 'C:\ProgramData\chocolatey\lib'
(i.e. C:\ProgramData\chocolatey\lib\yourPackageName).
A shim file for the command line goes to 'C:\ProgramData\chocolatey\bin'
and points to an executable in 'C:\ProgramData\chocolatey\lib\yourPackageName'.
Creating Chocolatey folders if they do not already exist.

chocolatey.nupkg file not installed in lib.
Attempting to locate it from bootstrapper.
PATH environment variable does not have C:\ProgramData\chocolatey\bin in it. Adding...
WARNING: Not setting tab completion: Profile file does not exist at
'C:\Users\Administrator\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1'.
Chocolatey (choco.exe) is now ready.
You can call choco from anywhere, command line or powershell by typing choco.
Run choco /? for a list of functions.
You may need to shut down and restart powershell and/or consoles
first prior to using choco.
Ensuring Chocolatey commands are on the path
Ensuring chocolatey.nupkg is in the lib folder
PS C:\Users\Administrator> choco list
Chocolatey v2.2.2
Chocolatey 2.2.2
1 packages installed.
PS C:\Users\Administrator>
```

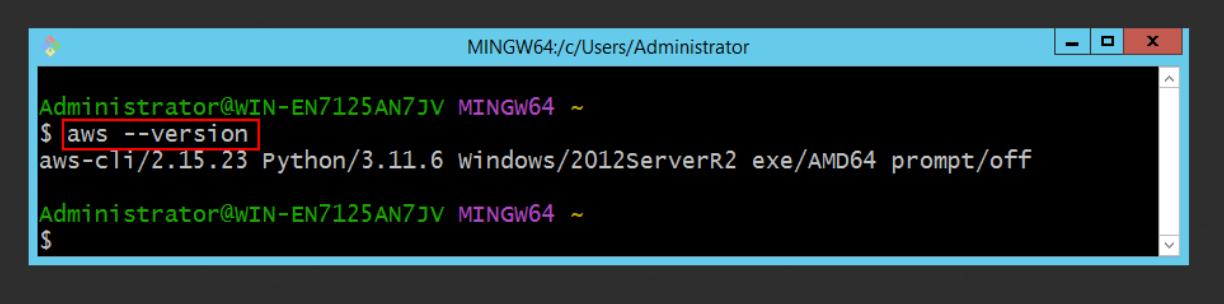
After installing **Chocolatey** in the **PowerShell** environment, you need to install the **AWS CLI** tool using the following command



```
Administrator: Windows PowerShell
PS C:\Users\Administrator> choco install awscli -y
chocolatey v2.2.2
Installing the following packages:
awscli
By installing, you accept licenses for the packages.
Progress: Downloading awscli 2.15.23... 100%
awscli v2.15.23 [Approved]
awscli package files install completed. Performing other installation steps.
Downloading awscli 64 bit
  from 'https://awscli.amazonaws.com/AWSCLIV2-2.15.23.msi'
Progress: 100% - Completed download of C:\Users\Administrator\AppData\Local\Temp\chocolatey\awscli\2.15.23\AWSCLIV2-2.15.23.msi (38.63 MB).
Download of AWSCLIV2-2.15.23.msi (38.63 MB) completed.
Hashes match
Installing awscli...
awscli has been installed.
awscli may be able to be automatically uninstalled.
Environment vars (like PATH) have changed. Close/reopen your shell to see the changes (or in powershell/cmd.exe just type `refreshenv`).
The install of awscli was successful.
Software installed as 'MSI', install location is likely default.

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\Users\Administrator>
```

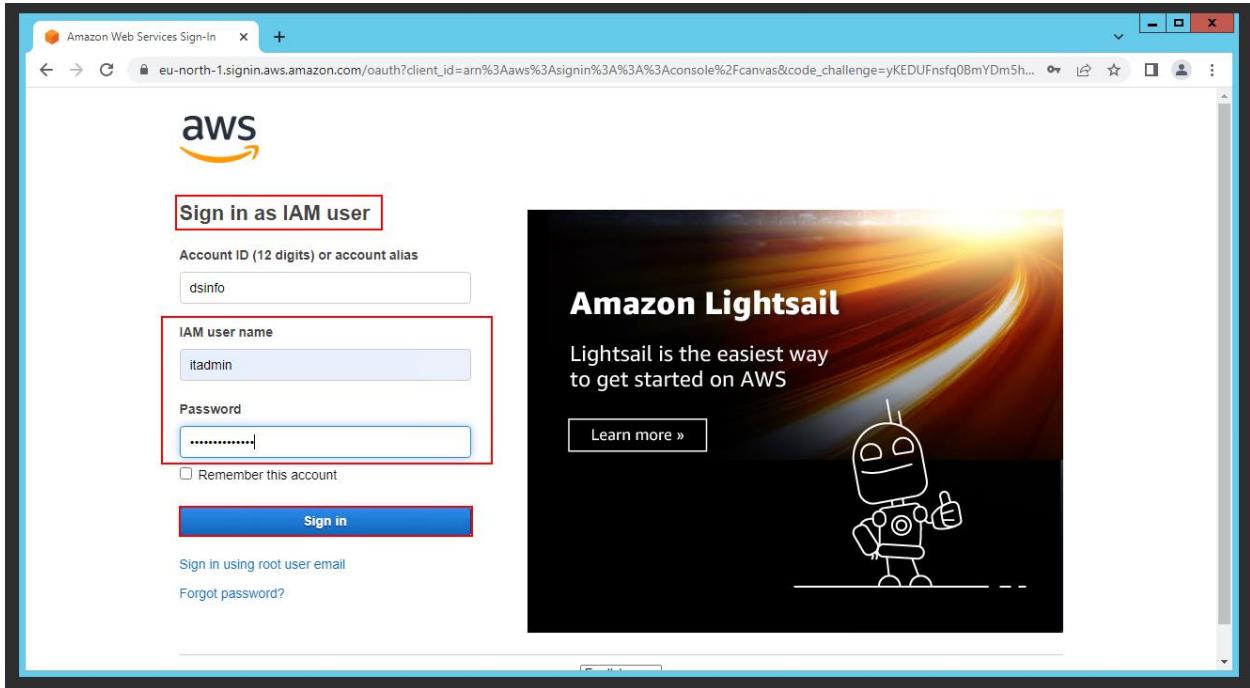
To check if **AWS CLI** has been installed correctly, use the following command in the **Git Bash** environment



```
Administrator@WIN-EN7125AN7JV MINGW64 ~
$ aws --version
aws-cli/2.15.23 Python/3.11.6 Windows/2012ServerR2 exe/AMD64 prompt/off

Administrator@WIN-EN7125AN7JV MINGW64 ~
$
```

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.

The screenshot shows the AWS Console Home page. On the left, there's a sidebar with links like EC2 and IAM. The IAM link is highlighted with a red box. The main area has four cards: 'Recently visited' (with EC2 and IAM listed), 'Applications' (empty), 'Welcome to AWS' (with links to Getting started with AWS and Training and certification), and 'Cost and usage' (showing 0 open issues and scheduled changes, with a prominent 'Access denied' message in a red box). At the bottom, there are buttons for 'View all services' and 'Go to myApplications'.

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

The screenshot shows the IAM Users page. The left sidebar has 'Users' highlighted with a red box. The main area shows a table of users with one entry: 'itadmin'. The table includes columns for User name, Path, Groups, Last activity, MFA, Password age, Console last sign-in, Access key ID, and Active key age. At the top right of the table, there are buttons for 'Create user' (highlighted with a red box) and 'Delete'.

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

Policy name	Type	Attached entities
AccessAnalyzerServiceRolePolicy	AWS managed	0
AdministratorAccess	AWS managed - job function	1
AdministratorAccess-Amplify	AWS managed	0
AdministratorAccess-AWSElasticBeanstalk	AWS managed	0
AlexaForBusinessDeviceSetup	AWS managed	0
AlexaForBusinessFullAccess	AWS managed	0
AlexaForBusinessGatewayExecution	AWS managed	0
AlexaForBusinessLifesizeDelegatedAccessPolicy	AWS managed	0

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

The screenshot shows the 'Review and create' step of the IAM 'Create user' wizard. It displays the user details and permissions summary for a user named 'awscli'. The user has 'None' as the console password type and 'No' as the require password reset setting. In the permissions summary, 'AdministratorAccess' is listed as an AWS managed - job function. There is a section for optional tags, which is currently empty. At the bottom right, there are 'Cancel', 'Previous', and 'Create user' buttons, with 'Create user' being highlighted.

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.

The screenshot shows the IAM user details page for 'awscli'. On the left is a navigation sidebar with options like 'Identity and Access Management (IAM)', 'Dashboard', 'Access management', 'Access reports', and 'Related consoles'. The main area shows the user's ARN, creation date, and access status. The 'Security credentials' tab is selected. Under 'Console sign-in', there is a link to the AWS console sign-in page. Under 'Multi-factor authentication (MFA)', there is a note about enabling MFA. Under 'Access keys', there is a note about using access keys and a red-bordered 'Create access key' button.

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

The screenshot shows the 'Access key best practices & alternatives' step of the AWS IAM Access Key creation wizard. On the left, a sidebar lists steps: Step 1 (Access key best practices & alternatives), Step 2 (optional) Set description tag, and Step 3 (optional) Retrieve access keys. The main content area has a title 'Access key best practices & alternatives' with a 'Info' link. It says 'Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.' Below this is a 'Use case' section with several options: 'Command Line Interface (CLI)' (selected, highlighted with a red border), 'Local code', 'Application running on an AWS compute service', 'Third-party service', 'Application running outside AWS', and 'Other'. Underneath is a 'Alternatives recommended' section with two bullet points: 'Use AWS CloudShell, a browser-based CLI, to run commands.' and 'Use the AWS CLI V2 and enable authentication through a user in IAM Identity Center.' At the bottom is a 'Confirmation' section with a checked checkbox 'I understand the above recommendation and want to proceed to create an access key.' and 'Cancel' and 'Next' buttons.

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

The screenshot shows the 'Set description tag - optional' step of the AWS IAM Access Key creation wizard. The sidebar shows steps: Step 1 (Access key best practices & alternatives), Step 2 (optional) Set description tag (selected), and Step 3 (optional) Retrieve access keys. The main content area has a title 'Set description tag - optional' with a 'Info' link. It says 'The description for this access key will be attached to this user as a tag and shown alongside the access key.' Below is a 'Description tag value' section with a text input field containing 'awscli'. A note says 'Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ / + - @'. At the bottom are 'Cancel', 'Previous', and 'Create access key' buttons, with 'Create access key' being highlighted.

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.

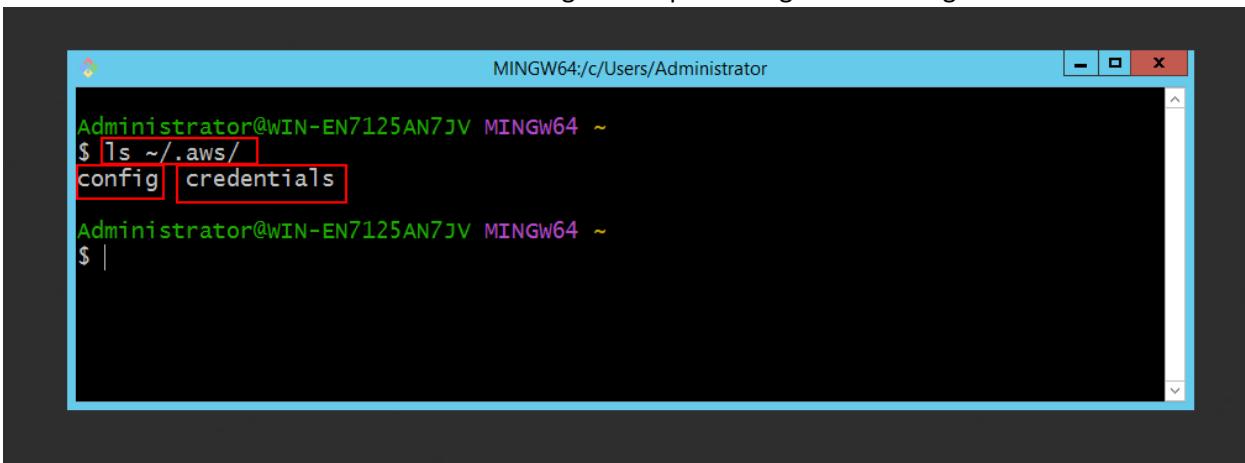
The screenshot shows the 'Create access key' step in the AWS IAM 'Retrieve access keys' wizard. It displays the 'Access key' and 'Secret access key' fields, both of which are highlighted with red boxes. Below these fields is a 'Access key best practices' section containing several bullet points about managing access keys. At the bottom right of the page are two buttons: 'Download .csv file' and 'Done'.

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**

The screenshot shows a terminal window titled 'MINGW64/c/Users/Administrator'. The user runs the command '\$ aws configure'. The terminal displays the configuration options:

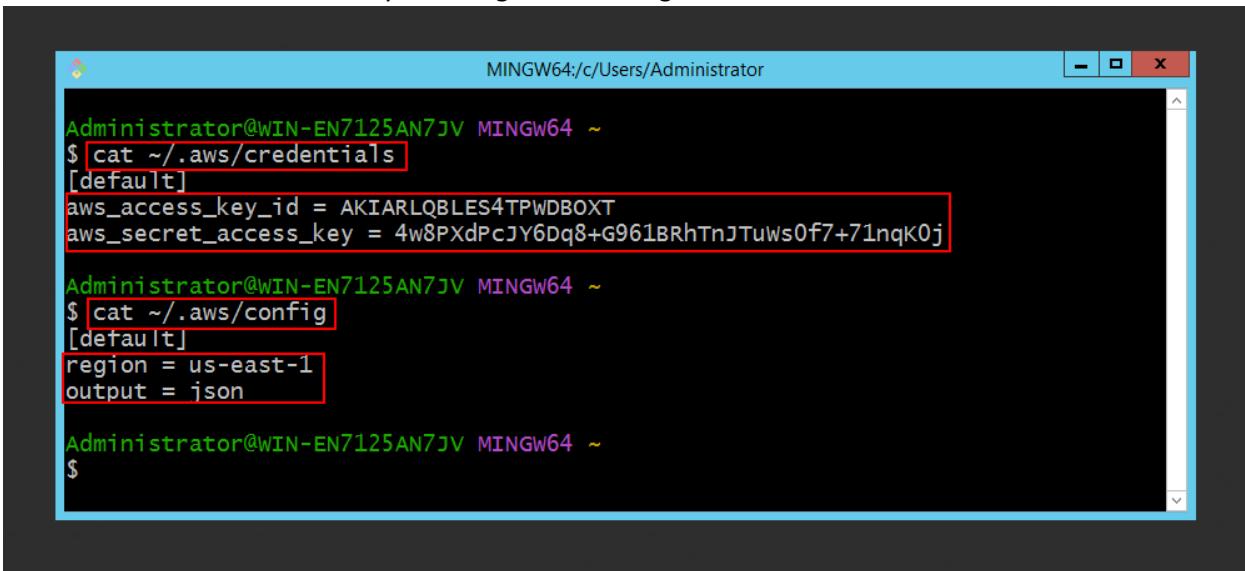
```
Administrator@WIN-EN7125AN7JV MINGW64 ~
$ aws configure
AWS Access Key ID [None]: AKIARLQBLES4TPWDBOXT
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
```

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



```
Administrator@WIN-EN7125AN7JV MINGW64 ~
$ ls ~/.aws/
config credentials
```

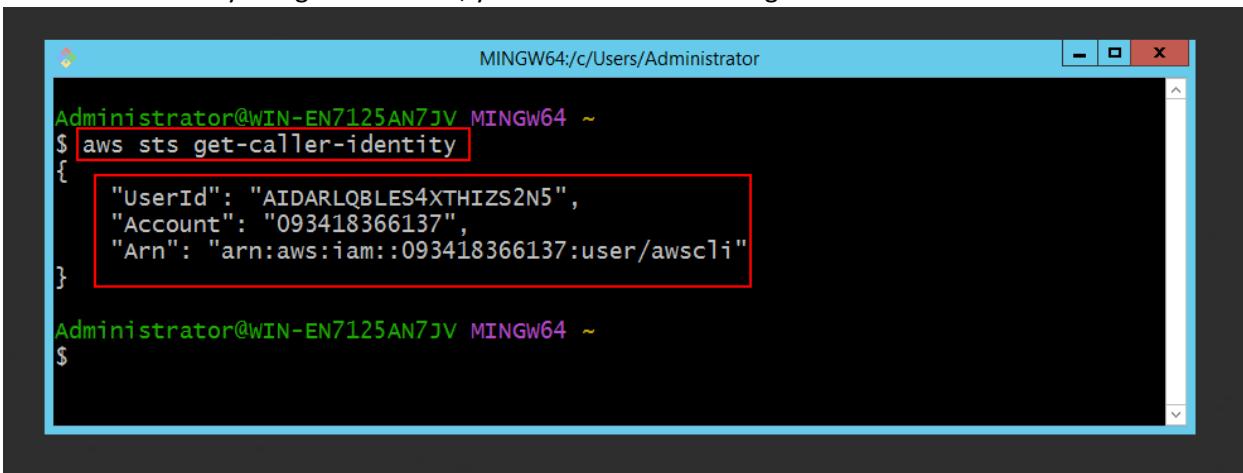
You can view the contents of any file using the following command



```
Administrator@WIN-EN7125AN7JV MINGW64 ~
$ cat ~/.aws/credentials
[default]
aws_access_key_id = AKIARLQBLES4TPWDBOXt
aws_secret_access_key = 4w8PxPcJY6Dq8+G961BRhTnJTuwS0f7+71nqk0j

Administrator@WIN-EN7125AN7JV MINGW64 ~
$ cat ~/.aws/config
[default]
region = us-east-1
output = json
```

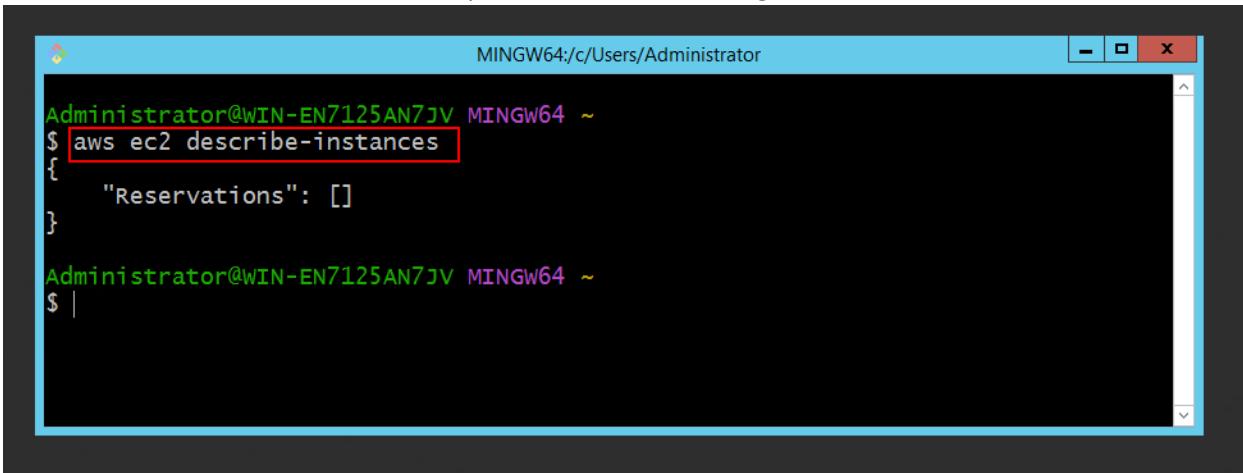
To view the identity using the AWS CLI, you can use the following command



```
Administrator@WIN-EN7125AN7JV MINGW64 ~
$ aws sts get-caller-identity
{
    "UserId": "AIDARLQBLES4XTHIZS2N5",
    "Account": "093418366137",
    "Arn": "arn:aws:iam::093418366137:user/awscli"
}

Administrator@WIN-EN7125AN7JV MINGW64 ~
$
```

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



```
Administrator@WIN-EN7125AN7JV MINGW64 ~
$ aws ec2 describe-instances
{
    "Reservations": []
}

Administrator@WIN-EN7125AN7JV MINGW64 ~
$ |
```

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**.

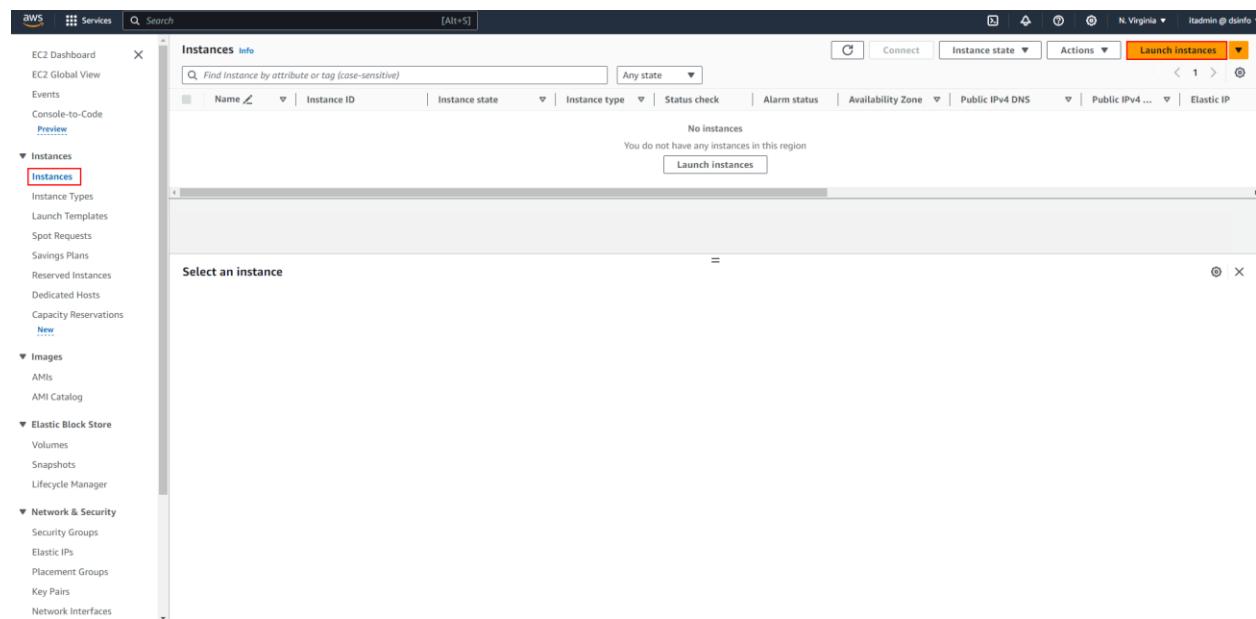
How to Create an EBS Volume

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**.

The screenshot shows the AWS EC2 'Launch an instance' wizard. In the 'Name and tags' step, the 'Name' field contains 'web01'. The 'Summary' pane on the right shows the instance configuration: 1 instance, Software Image (AMI) as Amazon Linux 2023.5.2..., Virtual server type as t2.micro, and 1 volume(s) - 8 GiB. A tooltip for the Free tier indicates it includes 750 hours of t2.micro or t3.micro usage in regions where t2.micro is available. The 'Launch instance' button is highlighted.

Step 3:

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

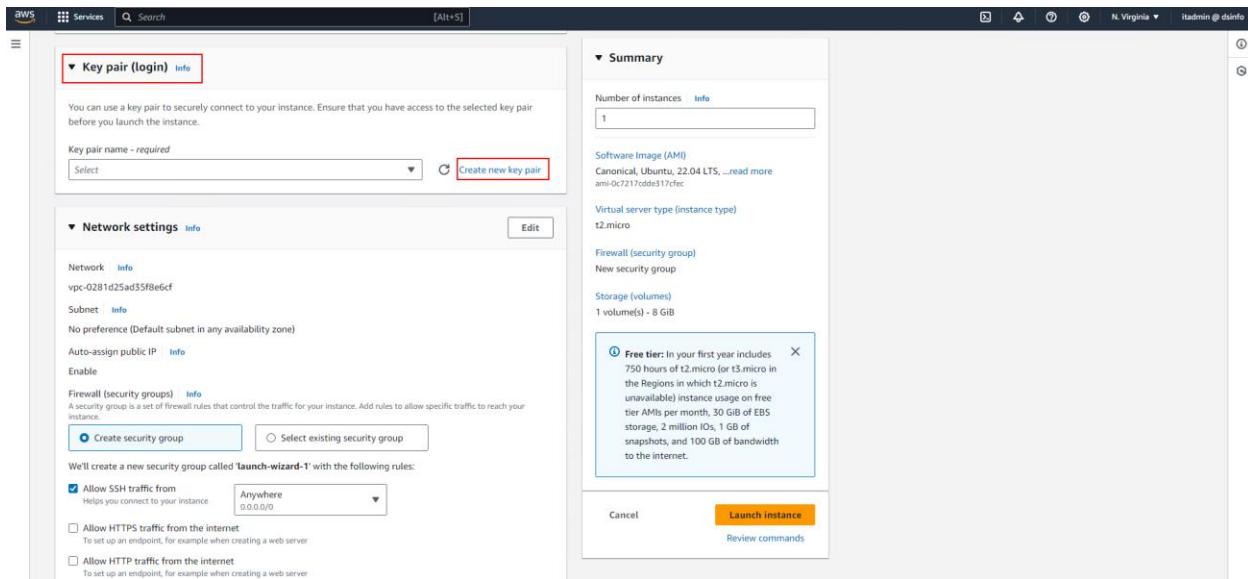
The screenshot shows the AWS Services dashboard with the search bar set to "Application and OS Images (Amazon Machine Image)". A search query "Ubuntu" has been entered. The results page displays various AMI categories: Ubuntu, Windows, Red Hat, SUSE Linux, and Debian. The "Ubuntu" category is selected, highlighted with a red border. On the right side of the screen, a "Summary" panel is open, showing the selected AMI: Canonical, Ubuntu, 22.04 LTS. The "Virtual server type (instance type)" dropdown is set to "t3.small". The "Launch instance" button is prominently displayed at the bottom right of the summary panel.

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

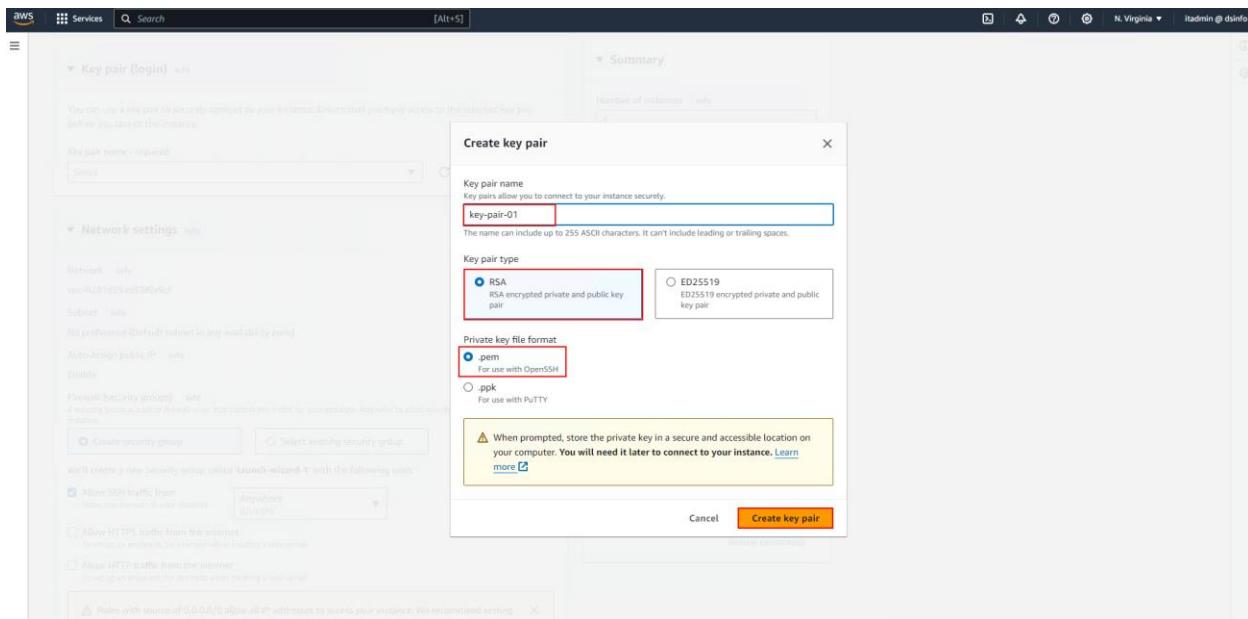
The screenshot shows the same AWS Services dashboard and search results for "Ubuntu" AMIs. In the "Instance type" section, the "t2.micro" option is selected and highlighted with a red border. This section also includes details about the instance type: Family: t2, 1 vCPU, 1 GB Memory, Current generation: true, and pricing information for On-Demand and On-Demand Reservated instances. The "Launch instance" button is visible at the bottom right.

Step 4:

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

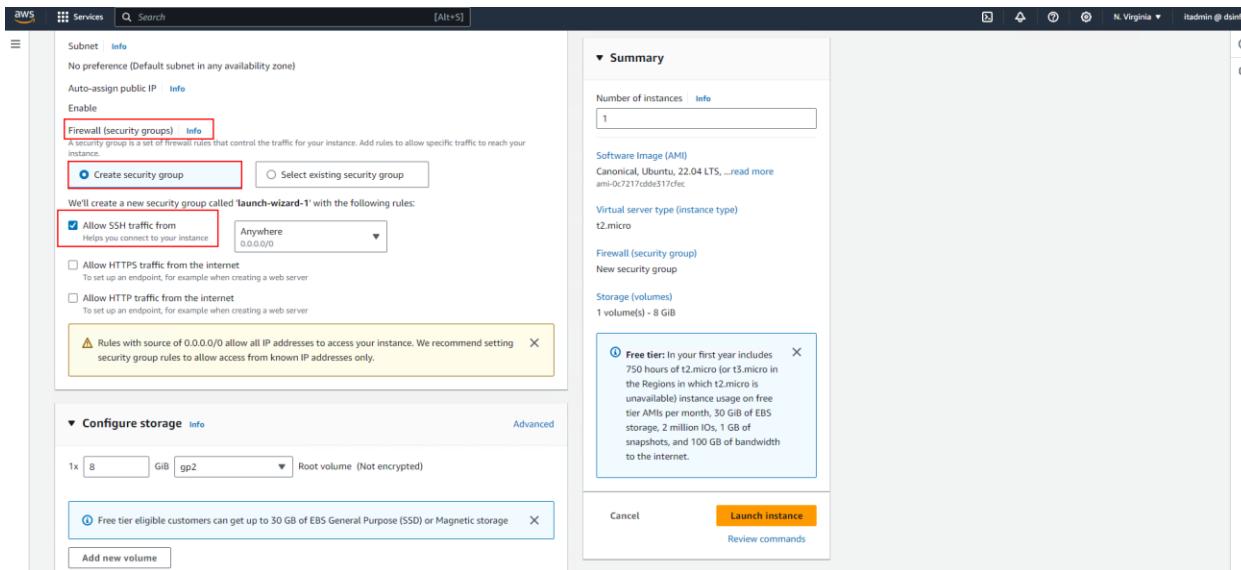


At this stage, create a Key Pair with the following specifications.

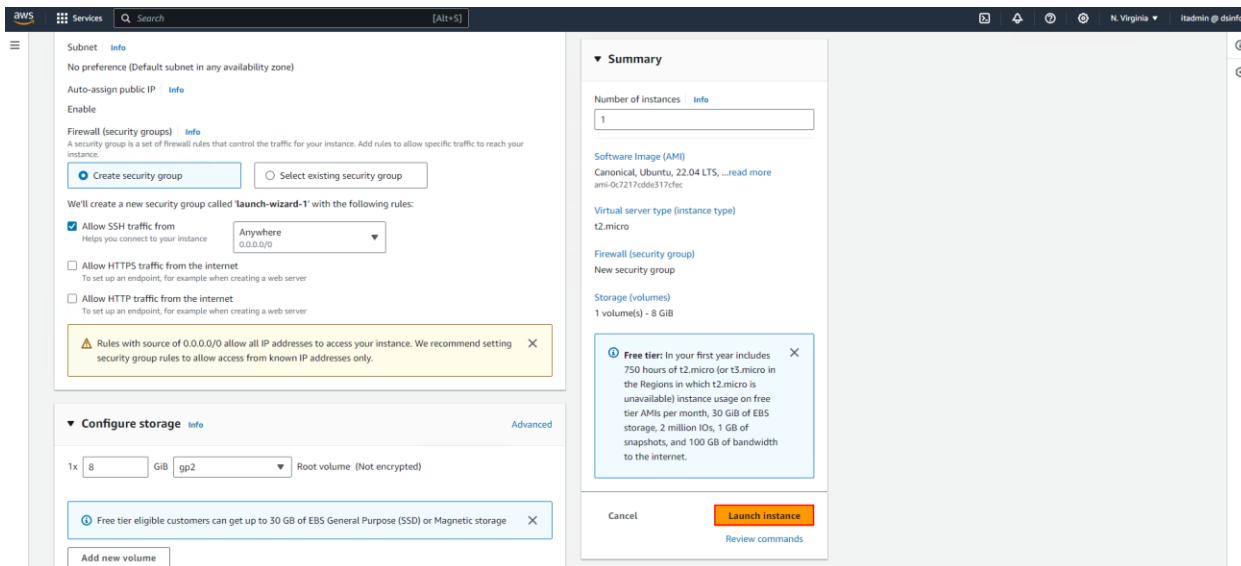


Step 5:

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

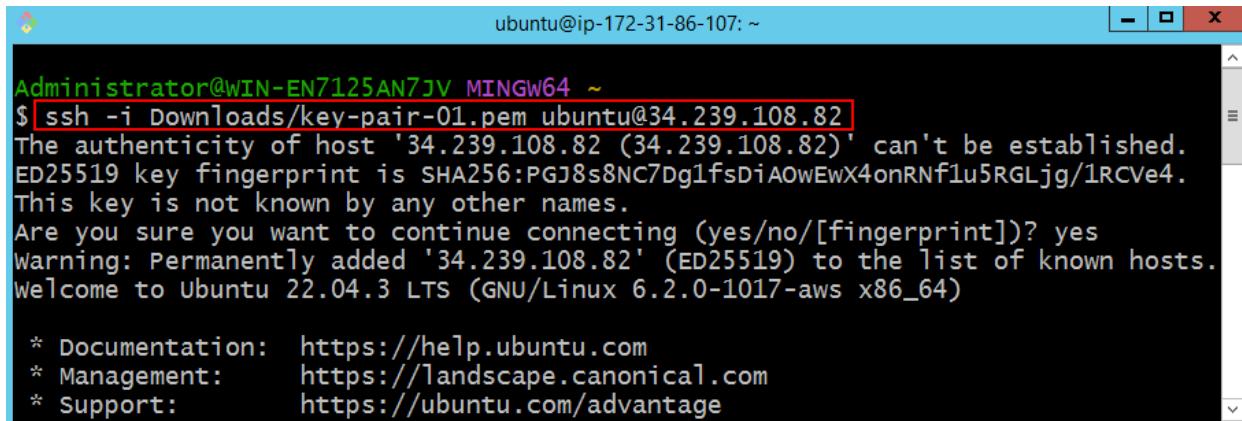


Then, click on the **Launch Instance** button to create the instance.



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.



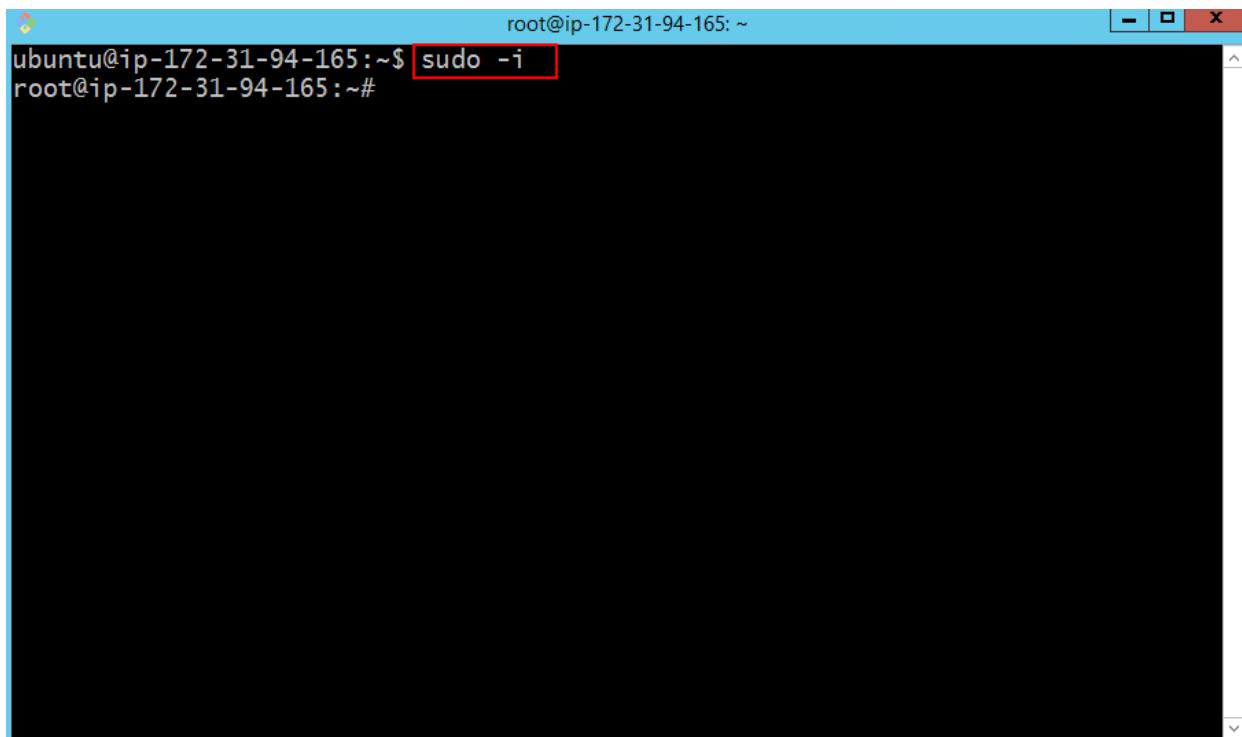
A screenshot of a terminal window titled "ubuntu@ip-172-31-86-107: ~". The window shows the command \$ ssh -i Downloads/key-pair-01.pem ubuntu@34.239.108.82 being entered. A red box highlights the command. The terminal then displays a warning about host fingerprint authentication, followed by a "yes" response, and a message indicating the host has been added to the list of known hosts. It then welcomes the user to Ubuntu 22.04.3 LTS. At the bottom, it provides documentation, management, and support links.

```
Administrator@WIN-EN7125AN7JV MINGW64 ~
$ ssh -i Downloads/key-pair-01.pem ubuntu@34.239.108.82
The authenticity of host '34.239.108.82 (34.239.108.82)' can't be established.
ED25519 key fingerprint is SHA256:PGJ8s8NC7Dg1fsDiaOwEwX4onRNf1u5RGLjg/1RCVe4.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '34.239.108.82' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1017-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage
```

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

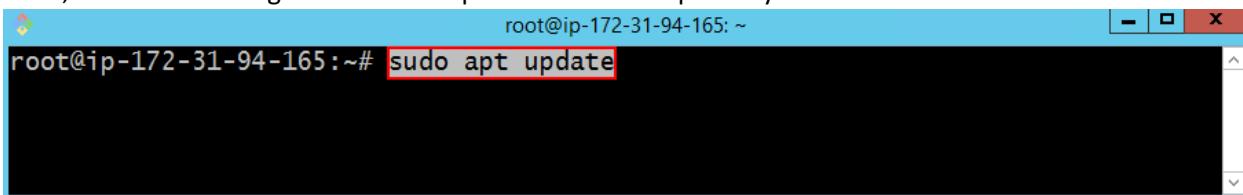
At this stage, use the following command to switch to the root user environment



A screenshot of a terminal window titled "root@ip-172-31-94-165: ~". The window shows the command \$ sudo -i being entered. A red box highlights the command. The terminal then displays a root prompt, indicating the user is now running as root.

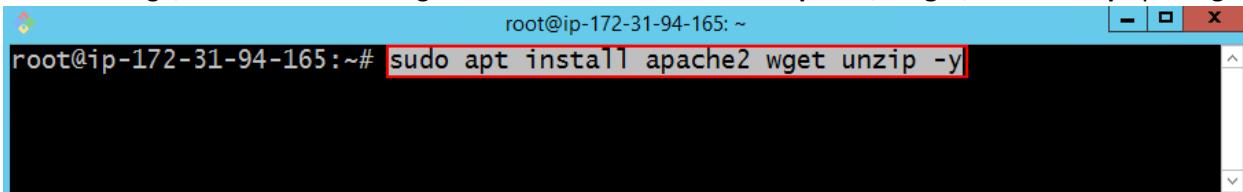
```
root@ip-172-31-94-165:~$ sudo -i
root@ip-172-31-94-165:~#
```

Then, use the following command to update the Linux repository



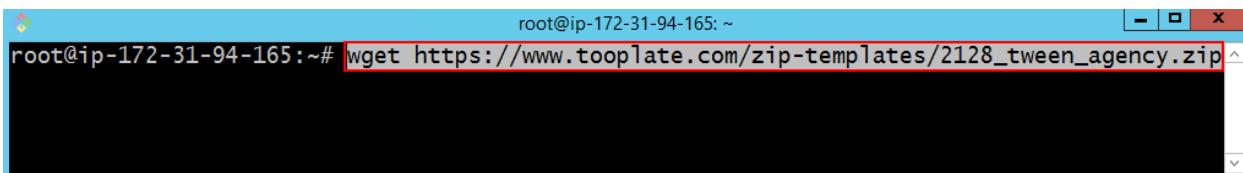
```
root@ip-172-31-94-165:~# sudo apt update
```

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



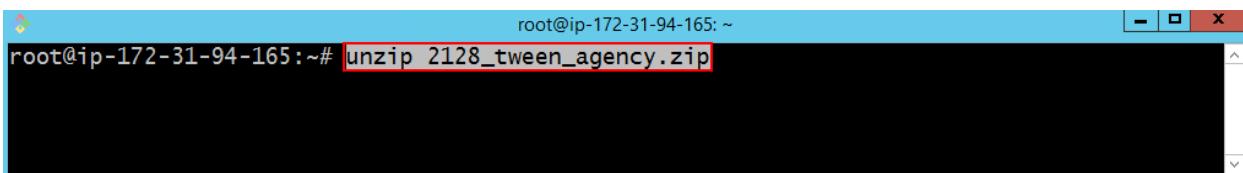
```
root@ip-172-31-94-165:~# sudo apt install apache2 wget unzip -y
```

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



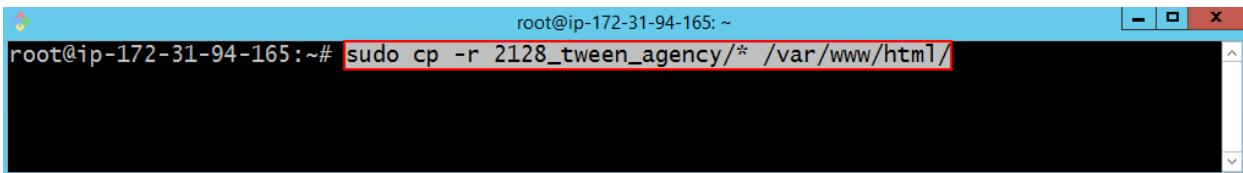
```
root@ip-172-31-94-165:~# wget https://www.tooplate.com/zip-templates/2128_tween_agency.zip
```

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



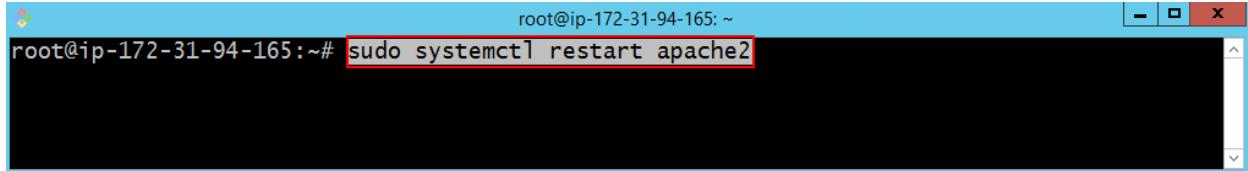
```
root@ip-172-31-94-165:~# unzip 2128_tween_agency.zip
```

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



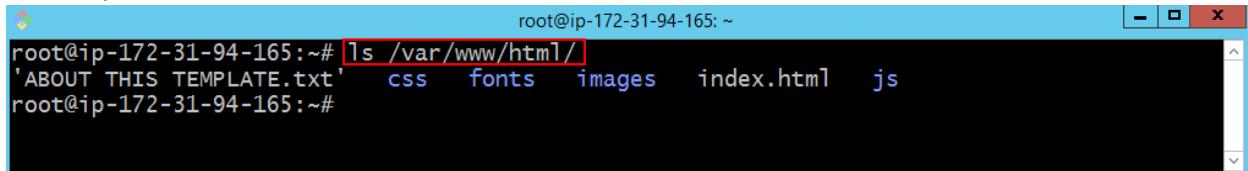
```
root@ip-172-31-94-165:~# sudo cp -r 2128_tween_agency/* /var/www/html/
```

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



```
root@ip-172-31-94-165:~# sudo systemctl restart apache2
```

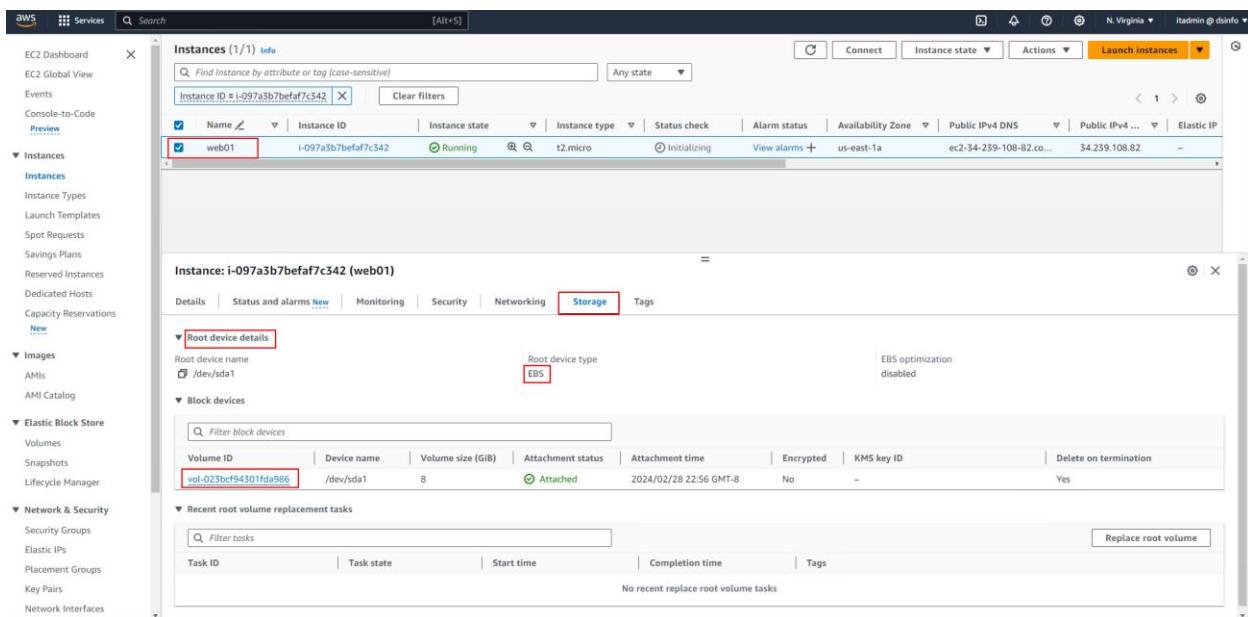
Use the following command to check if the **Tween** website files have been copied to the web server directory



```
root@ip-172-31-94-165:~# ls /var/www/html/
'ABOUT THIS TEMPLATE.txt'  css  fonts  images  index.html  js
root@ip-172-31-94-165:~#
```

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.



The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like EC2 Dashboard, Services, Events, and various AWS services. The main area shows one instance named 'web01' with the following details:

Details	Status and alarms	Monitoring	Security	Networking	Storage	Tags
Instance ID: i-097a3b7befaf7c342	New				Root device type: EBS	
Root device name: /dev/sda1						EBS optimization disabled
Block devices:						
Volume ID: vol-023bcf94301fd9986	Device name: /dev/sda1	Volume size (GiB): 8	Attachment status: Attached	Attachment time: 2024/02/28 22:56 GMT-8	Encrypted: No	KMS key ID: -
						Delete on termination: Yes

Below the instance details, there's a section for 'Recent root volume replacement tasks' with a 'Replace root volume' button.

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

The screenshot shows the AWS EC2 Volumes page. A modal dialog is open over the main table, allowing the user to edit the volume's name. The 'Name' field contains 'web01-root-volume'. The 'Save' button is visible at the bottom of the dialog.

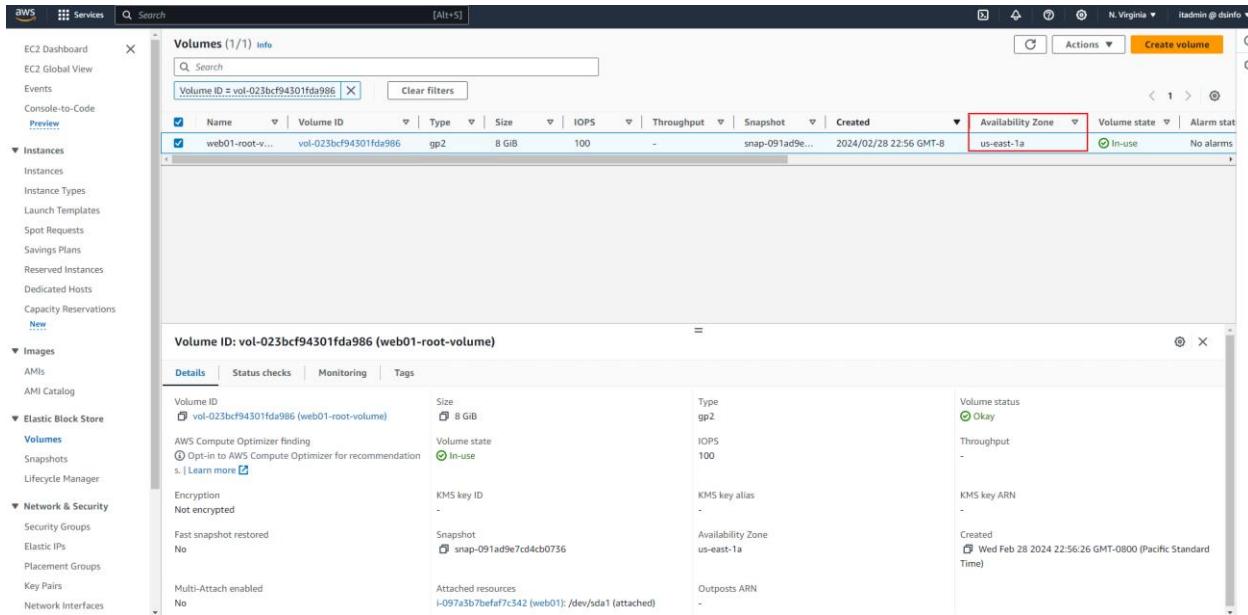
Name	Type	Size	IOPS	Throughput	Snapshot	Created	Availability Zone	Volume state	Alarm stat
gp2	8 GiB	100	-	-	snap-091ad9e...	2024/02/28 22:56 GMT-8	us-east-1a	In-use	No alarms

Volume ID: vol-023bcf94301fda986

Details | Status checks | Monitoring | Tags

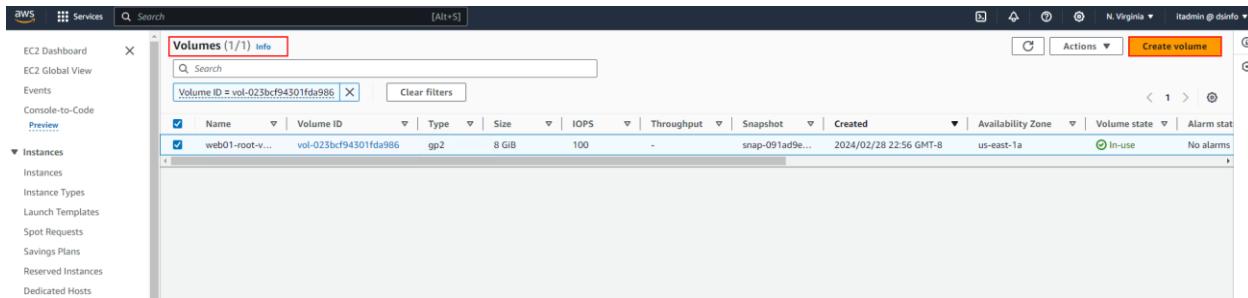
Volume ID vol-023bcf94301fda986	Size 8 GiB	Type gp2	Volume status Okay
AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendation s. Learn more	Volume state In-use	IOPS 100	Throughput
Encryption Not encrypted	KMS key ID	KMS key alias	KMS key ARN
Fast snapshot restored No	Snapshot snap-091ad9e7cd4cb0736	Availability Zone us-east-1a	Created Wed Feb 28 2024 22:56:26 GMT-0800 (Pacific Standard Time)
Multi-Attach enabled No	Attached resources i-097a3b7befaf7c542 (web01): /dev/sda1 (attached)	Outposts ARN	

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**.



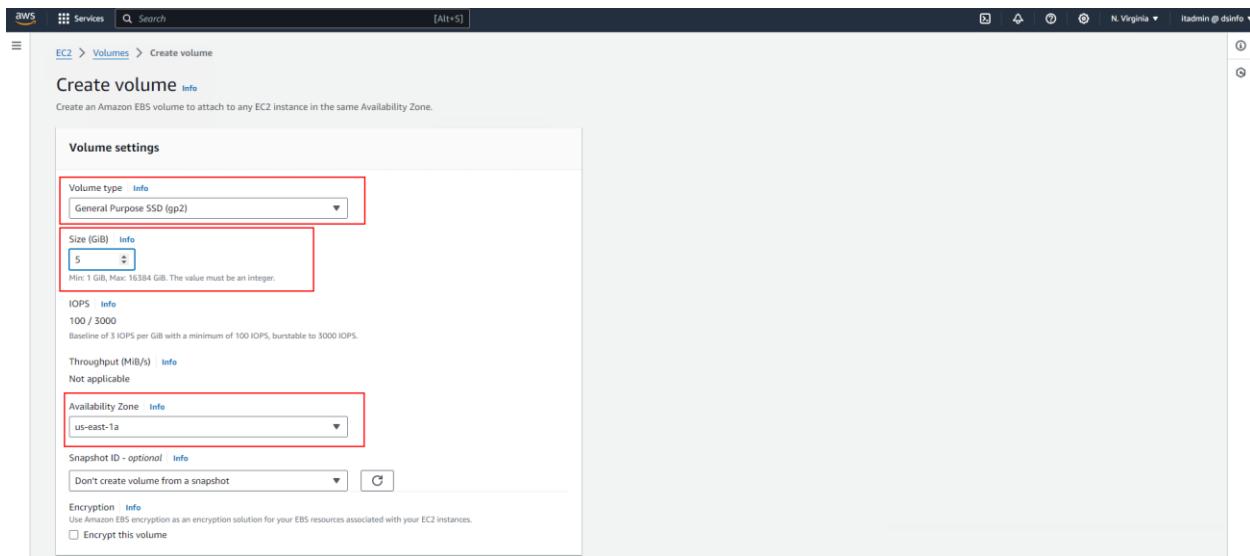
The screenshot shows the AWS EC2 Volumes page. On the left, there's a navigation sidebar with options like EC2 Dashboard, EC2 Global View, Events, Console-to-Code, Instances, Images, Elastic Block Store, Network & Security, and more. The main area is titled "Volumes (1/1) Info". It displays a table with one row for a volume named "web01-root-v...". The table includes columns for Name, Volume ID, Type, Size, IOPS, Throughput, Snapshot, Created, Availability Zone, Volume state, and Alarm stat. The "Availability Zone" column shows "us-east-1a", which is highlighted with a red border. The "Volume state" column shows "In-use". Below the table, there's a detailed view for the selected volume, showing its ID as "vol-023bcf94301fda986" and its name as "web01-root-volume". The details tab is active, displaying information such as Volume ID, Size (8 GiB), Type (gp2), AWS Compute Optimizer finding (Opt-in to AWS Compute Optimizer for recommendation), Encryption (Not encrypted), Fast snapshot restored (No), Multi-Attach enabled (No), and Attached resources (i-097a3b7befaf7c542 (web01): /dev/sda1 (attached)).

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

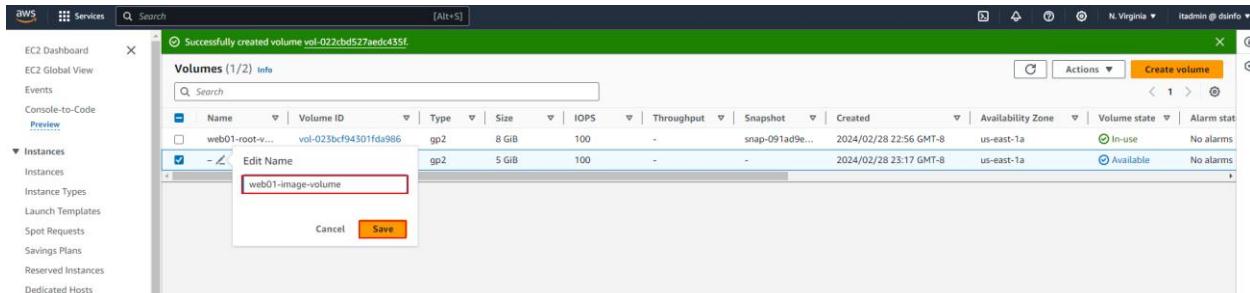


This screenshot is similar to the previous one, showing the AWS EC2 Volumes page. The main difference is that the "Create volume" button at the top right of the "Volumes (1/1) Info" section is highlighted with a red border. The rest of the interface, including the sidebar and the detailed volume view, appears identical to the first screenshot.

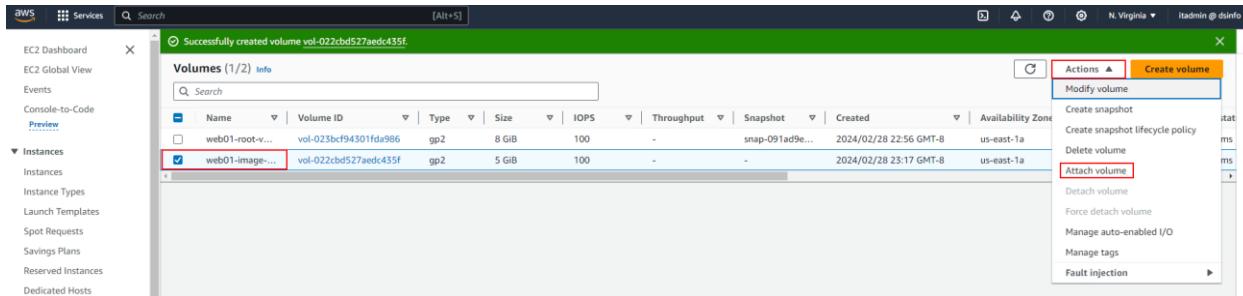
At this stage, select **General Purpose SSD (gp2)** from the **Volume Type** section, set the **Size** to **5GB**, and make sure to choose the same **Availability Zone** as your instance. Finally, click the **Create Volume** button.



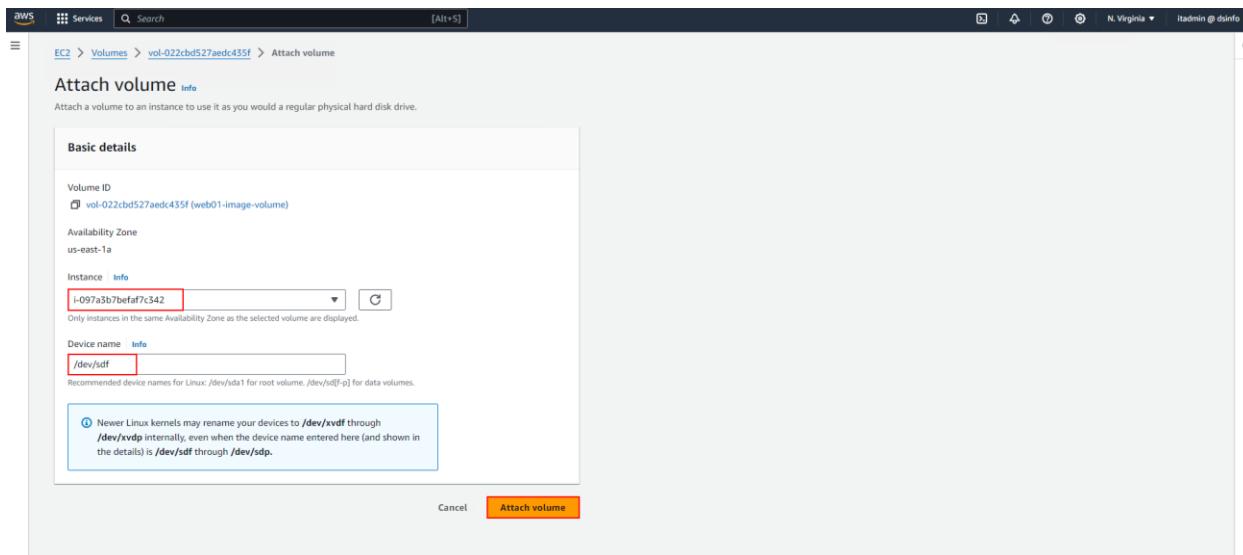
At this stage, assign a **name** to the EBS Volume.



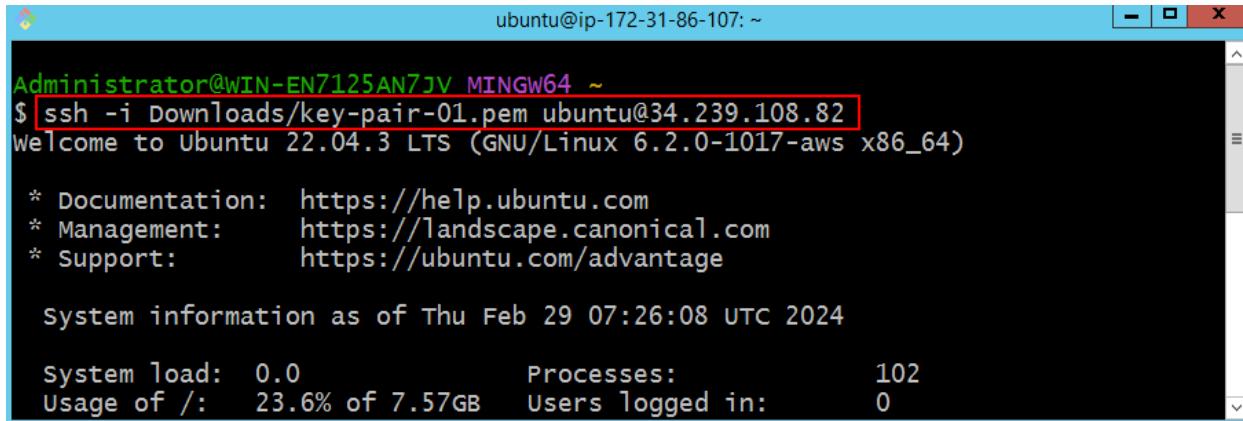
To attach the **EBS Volume** to the **EC2 Instance**, click on the **Actions** button and then select **Attach Volume**.



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.



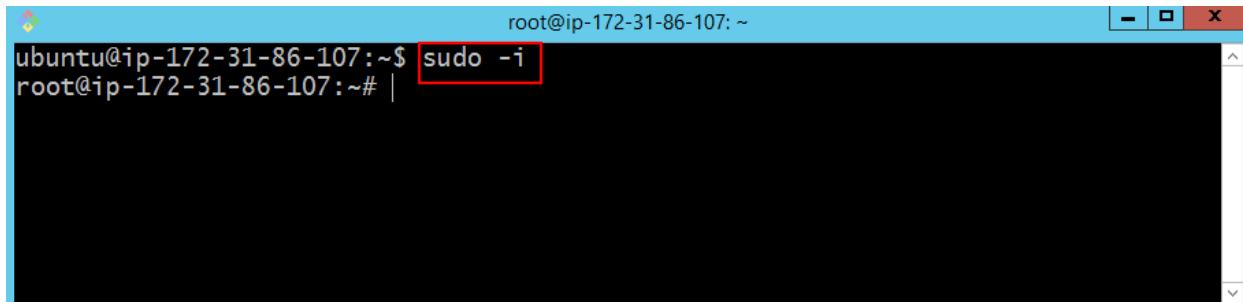
```
ubuntu@ip-172-31-86-107: ~
Administrator@WIN-EN7125AN7JV MINGW64 ~
$ ssh -i Downloads/key-pair-01.pem ubuntu@34.239.108.82
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1017-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

 System information as of Thu Feb 29 07:26:08 UTC 2024

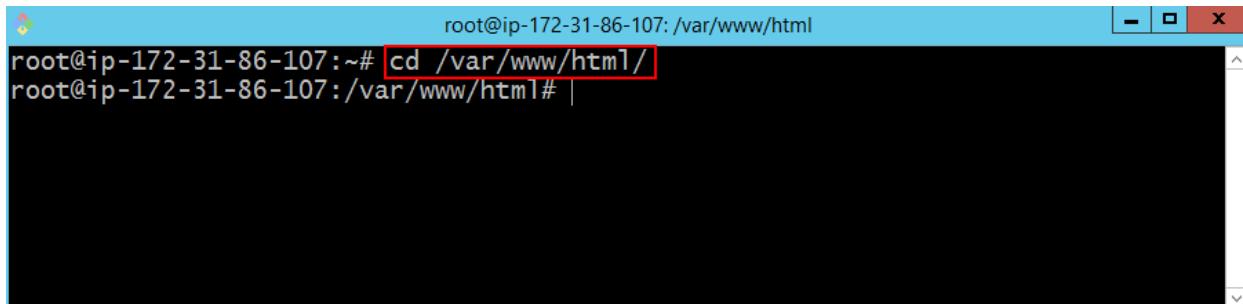
 System load: 0.0          Processes: 102
 Usage of /: 23.6% of 7.57GB   Users logged in: 0
```

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



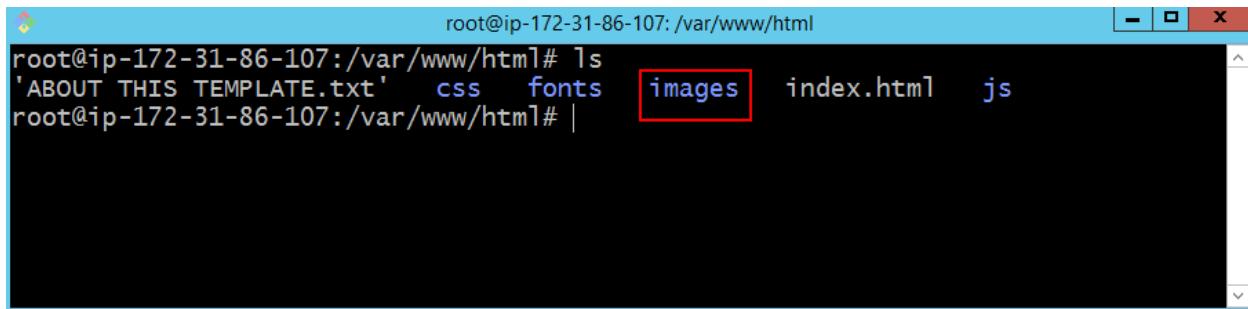
```
root@ip-172-31-86-107: ~
ubuntu@ip-172-31-86-107:~$ sudo -i
root@ip-172-31-86-107:~# |
```

Enter the **web server directory**



```
root@ip-172-31-86-107:~/var/www/html
root@ip-172-31-86-107:~/var/www/html# cd /var/www/html/
root@ip-172-31-86-107:/var/www/html# |
```

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



```
root@ip-172-31-86-107:/var/www/html# ls  
'ABOUT THIS TEMPLATE.txt'  css  fonts  images  index.html  js  
root@ip-172-31-86-107:/var/www/html# |
```

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

```
root@ip-172-31-86-107:/var/www/html# fdisk -l
Disk /dev/loop0: 24.9 MiB, 26112000 bytes, 51000 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop1: 55.66 MiB, 58363904 bytes, 113992 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop2: 63.46 MiB, 66547712 bytes, 129976 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop3: 111.95 MiB, 117387264 bytes, 229272 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop4: 40.86 MiB, 42840064 bytes, 83672 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

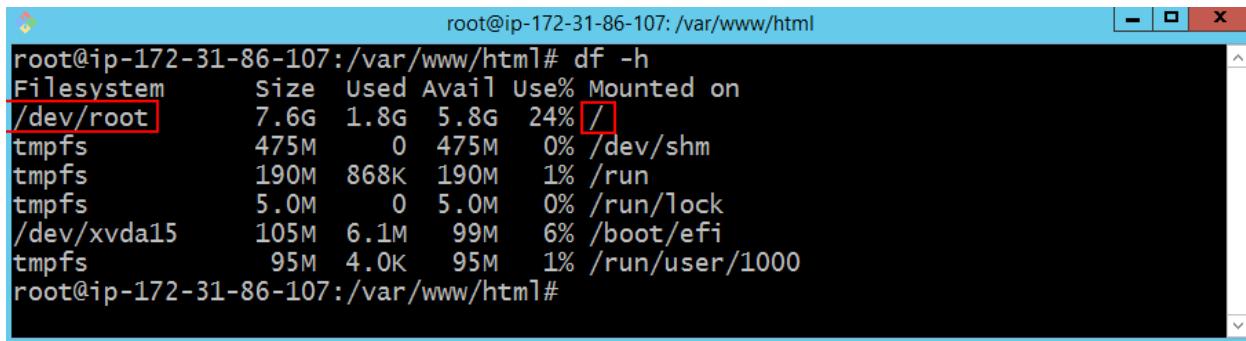
Disk /dev/xvda: 8 GiB, 8589934592 bytes, 16777216 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 8F814D15-0F5D-40E2-8A1E-E14BBDFCE594

Device      Start      End  Sectors  Size Type
/dev/xvda1   227328  16777182 16549855  7.9G Linux filesystem
/dev/xvda14    2048     10239     8192    4M BIOS boot
/dev/xvda15   10240    227327  217088 106M EFI System

Partition table entries are not in disk order.

Disk /dev/xvdf: 5 GiB, 5368709120 bytes, 10485760 sectors
Units: sectors of 1 * 512 = 512 bytes
```

Use the following command to view the mounted partitions



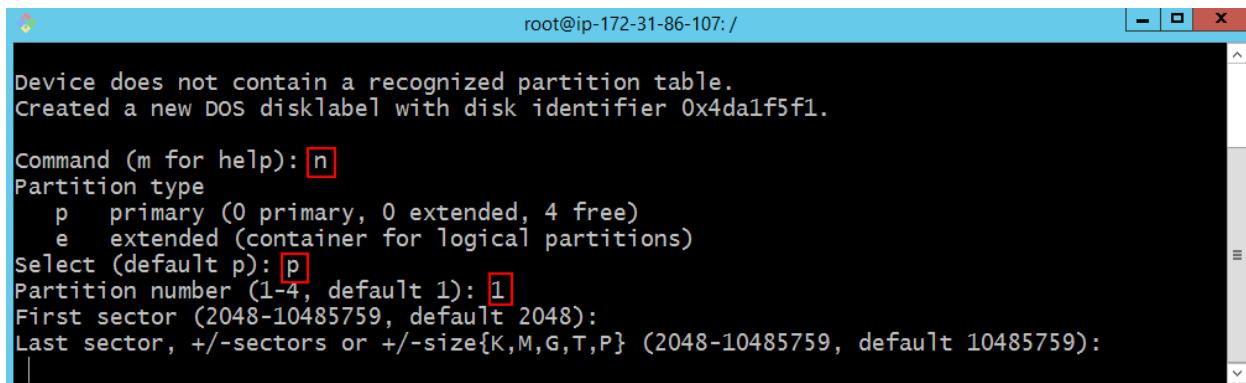
```
root@ip-172-31-86-107:/var/www/html# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       7.6G  1.8G  5.8G  24% /
tmpfs          475M    0  475M   0% /dev/shm
tmpfs          190M  868K  190M   1% /run
tmpfs          5.0M    0  5.0M   0% /run/lock
/dev/xvda15     105M  6.1M   99M   6% /boot/efi
tmpfs          95M  4.0K   95M   1% /run/user/1000
root@ip-172-31-86-107:/var/www/html#
```

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



```
root@ip-172-31-86-107:/# fdisk /dev/xvdf
```

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**.



```
Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x4da1f5f1.

Command (m for help): n
Partition type
  p  primary (0 primary, 0 extended, 4 free)
  e  extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-10485759, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-10485759, default 10485759):
```

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

```
root@ip-172-31-86-107: /  
Command (m for help): p  
Disk /dev/xvdf: 5 GiB, 5368709120 bytes, 10485760 sectors  
Units: sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disklabel type: dos  
Disk identifier: 0x4da1f5f1  
  
Device      Boot Start      End  Sectors Size Id Type  
/dev/xvdf1          2048 10485759 10483712  5G 83 Linux  
  
Command (m for help): |
```

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

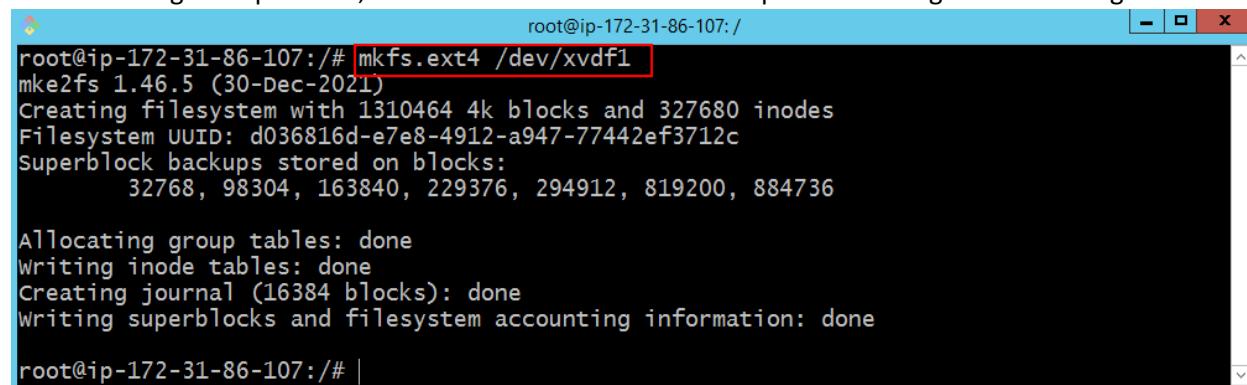
```
root@ip-172-31-86-107: /  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disklabel type: dos  
Disk identifier: 0x4da1f5f1  
  
Device      Boot Start      End  Sectors Size Id Type  
/dev/xvdf1          2048 10485759 10483712  5G 83 Linux  
  
Command (m for help): w  
The partition table has been altered.  
Calling ioctl() to re-read partition table.  
Syncing disks.  
root@ip-172-31-86-107: #
```

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

fdisk -l

```
root@ip-172-31-86-107: /  
Partition table entries are not in disk order.  
  
Disk /dev/xvdf: 5 GiB, 5368709120 bytes, 10485760 sectors  
Units: sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disklabel type: dos  
Disk identifier: 0x4da1f5f1  
  
Device      Boot Start      End  Sectors Size Id Type  
/dev/xvdf1          2048 10485759 10483712  5G 83 Linux  
root@ip-172-31-86-107: # |
```

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

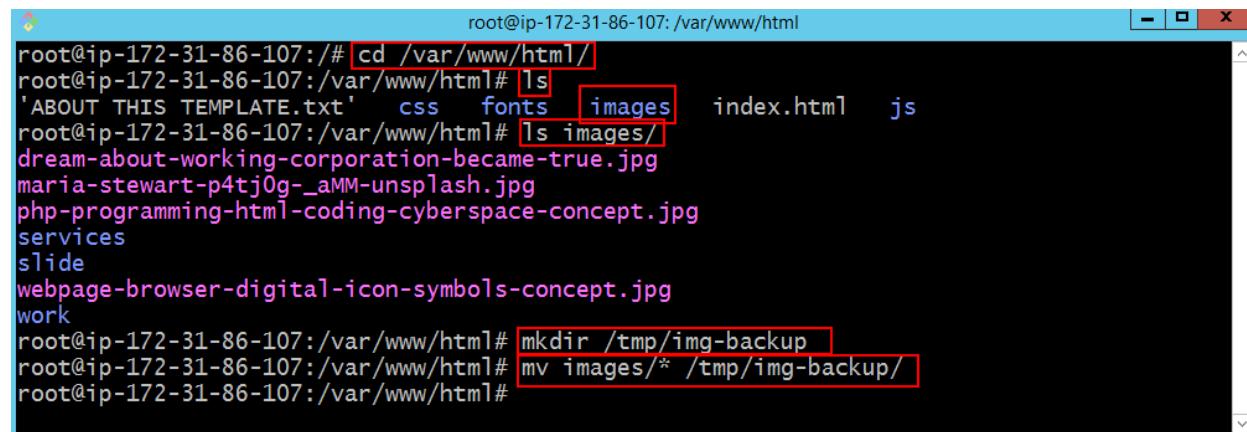


```
root@ip-172-31-86-107:/# mkfs.ext4 /dev/xvdf1
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 1310464 4k blocks and 327680 inodes
Filesystem UUID: d036816d-e7e8-4912-a947-77442ef3712c
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

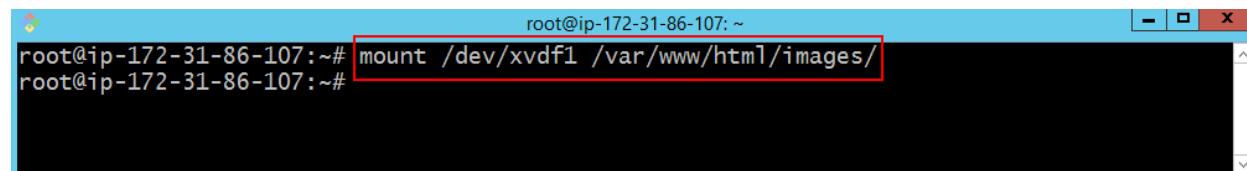
root@ip-172-31-86-107:/# |
```

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.



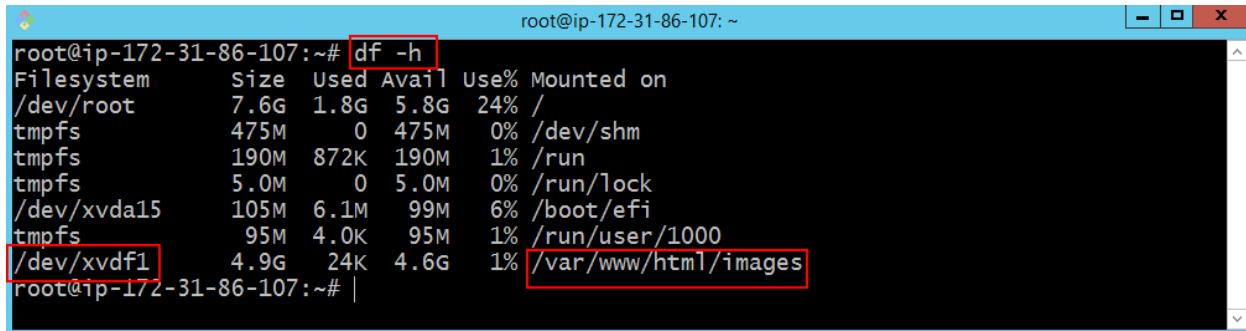
```
root@ip-172-31-86-107:/# cd /var/www/html/
root@ip-172-31-86-107:/var/www/html# ls
'ABOUT THIS TEMPLATE.txt'  css  fonts  images  index.html  js
root@ip-172-31-86-107:/var/www/html# ls images/
dream-about-working-corporation-became-true.jpg
maria-stewart-p4tj0g-_aMM-unsplash.jpg
php-programming-html-coding-cyberspace-concept.jpg
services
slide
webpage-browser-digital-icon-symbols-concept.jpg
work
root@ip-172-31-86-107:/var/www/html# mkdir /tmp/img-backup
root@ip-172-31-86-107:/var/www/html# mv images/* /tmp/img-backup/
root@ip-172-31-86-107:/var/www/html#
```

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



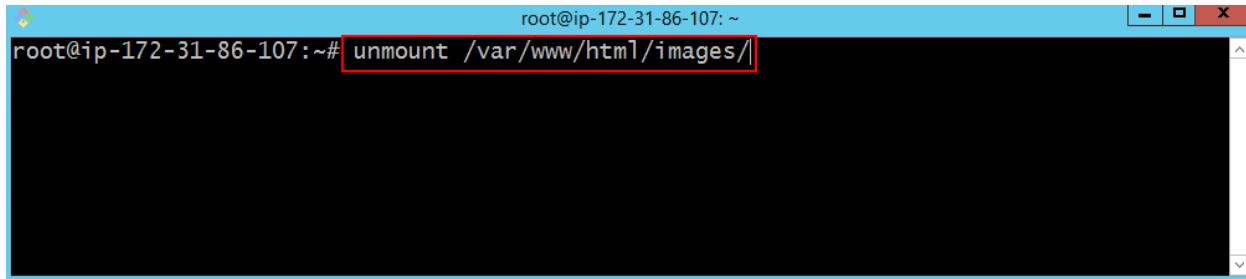
```
root@ip-172-31-86-107:~# mount /dev/xvdf1 /var/www/html/images/
root@ip-172-31-86-107:~#
```

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



```
root@ip-172-31-86-107:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       7.6G  1.8G  5.8G  24% /
tmpfs          475M    0  475M   0% /dev/shm
tmpfs          190M  872K  190M   1% /run
tmpfs           5.0M    0  5.0M   0% /run/lock
/dev/xvda15     105M  6.1M   99M   6% /boot/efi
tmpfs           95M  4.0K   95M   1% /run/user/1000
/dev/xvdf1      4.9G  24K  4.6G   1% /var/www/html/images/
root@ip-172-31-86-107:~#
```

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



```
root@ip-172-31-86-107:~#
root@ip-172-31-86-107:~# unmount /var/www/html/images/
```

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



```
root@ip-172-31-86-107:~#
root@ip-172-31-86-107:~# nano /etc/fstab
```

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

```
root@ip-172-31-86-107: ~
GNU nano 6.2
/etc/fstab *
LABEL=cloudimg-rootfs / ext4 discard,errors=remount-ro 0 1
LABEL=UEFI /boot/efi vfat umask=0077 0 1
/dev/xvdf1 [red box] /var/www/html/images ext4 default 0 0
[TAB KEY]
[red arrow]
[red arrow]
[red arrow]
No File System Dump
No File System Checking
```

File menu: ^G Help, ^O Write Out, ^W Where Is, ^K Cut, ^T Execute, ^C Location
Edit menu: ^X Exit, ^R Read File, ^V Replace, ^U Paste, ^J Justify, / Go To Line

Then, save the file.

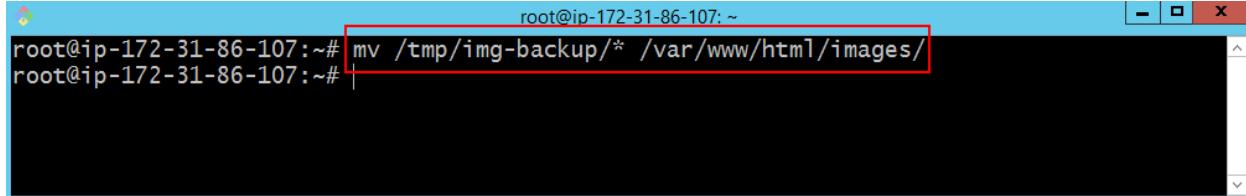
Use the following command to mount all entries defined in the **fstab** file

```
root@ip-172-31-86-107: ~# [red box] mount -a
root@ip-172-31-86-107: ~|
```

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

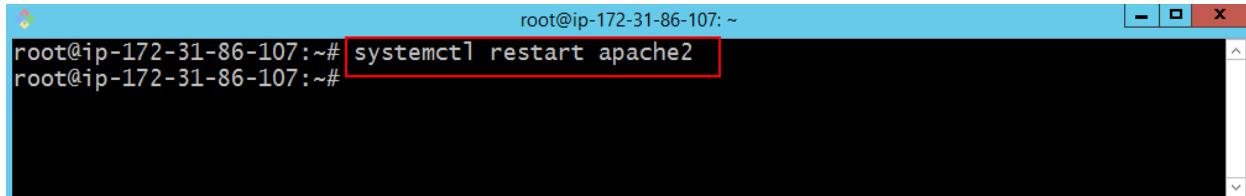
```
root@ip-172-31-86-107: ~# [red box] df -h
root@ip-172-31-86-107: ~#
Filesystem      Size  Used Avail Use% Mounted on
/dev/root      7.6G  1.8G  5.8G  24% /
tmpfs          475M    0  475M   0% /dev/shm
tmpfs          190M  872K 190M   1% /run
tmpfs          5.0M    0  5.0M   0% /run/lock
/dev/xvda15    105M  6.1M  99M   6% /boot/efi
tmpfs          95M  4.0K  95M   1% /run/user/1000
[red box] /dev/xvdf1  4.9G  24K  4.6G  1% [/var/www/html/images]
root@ip-172-31-86-107: ~|
```

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



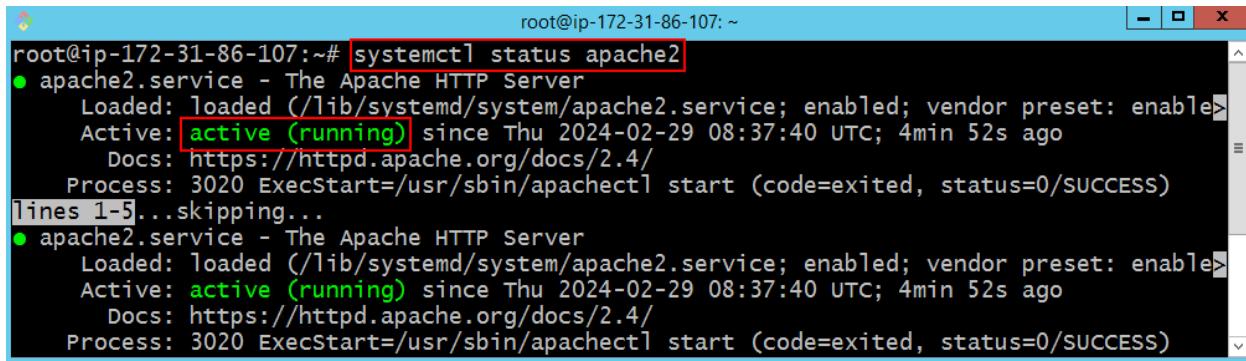
```
root@ip-172-31-86-107:~# mv /tmp/img-backup/* /var/www/html/images/
root@ip-172-31-86-107:~#
```

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



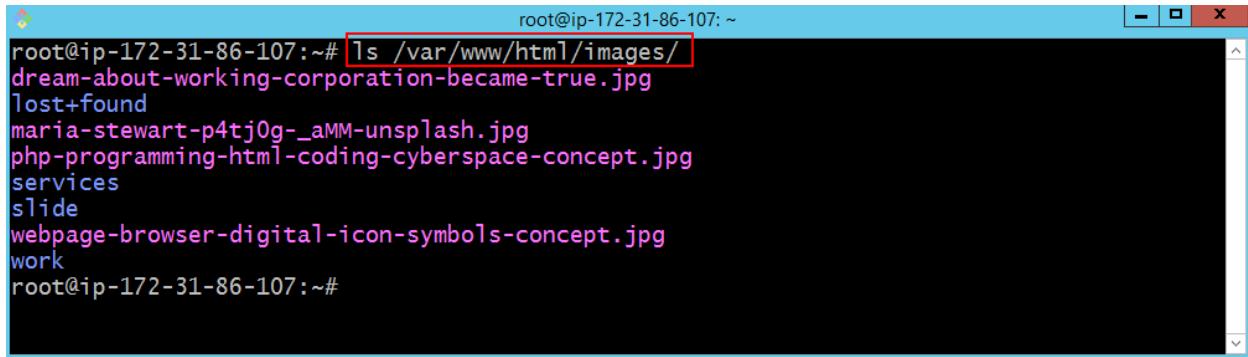
```
root@ip-172-31-86-107:~# systemctl restart apache2
root@ip-172-31-86-107:~#
```

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



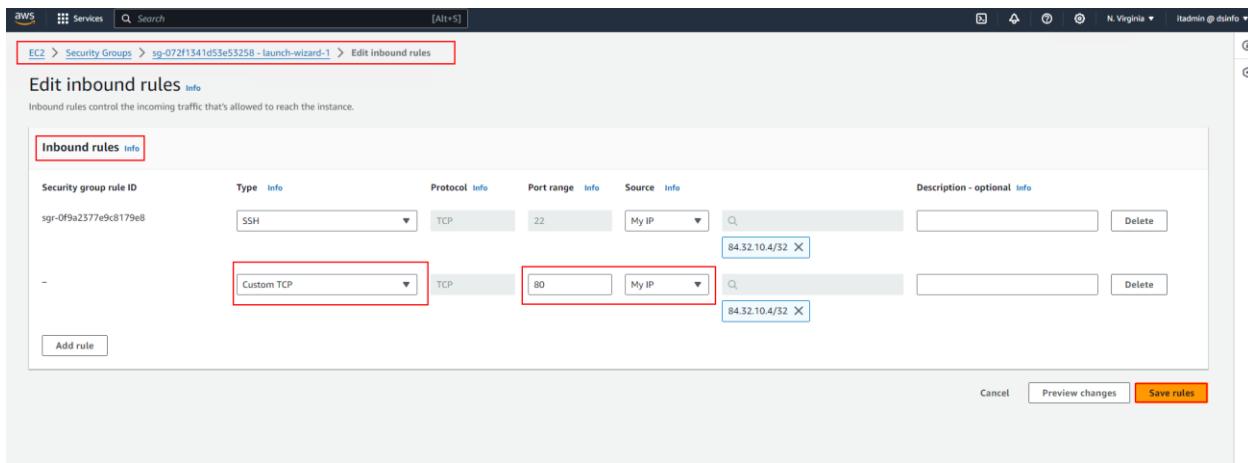
```
root@ip-172-31-86-107:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
    Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
    Active: active (running) since Thu 2024-02-29 08:37:40 UTC; 4min 52s ago
      Docs: https://httpd.apache.org/docs/2.4/
     Process: 3020 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
lines 1-5... skipping...
● apache2.service - The Apache HTTP Server
    Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
    Active: active (running) since Thu 2024-02-29 08:37:40 UTC; 4min 52s ago
      Docs: https://httpd.apache.org/docs/2.4/
     Process: 3020 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
```

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**.

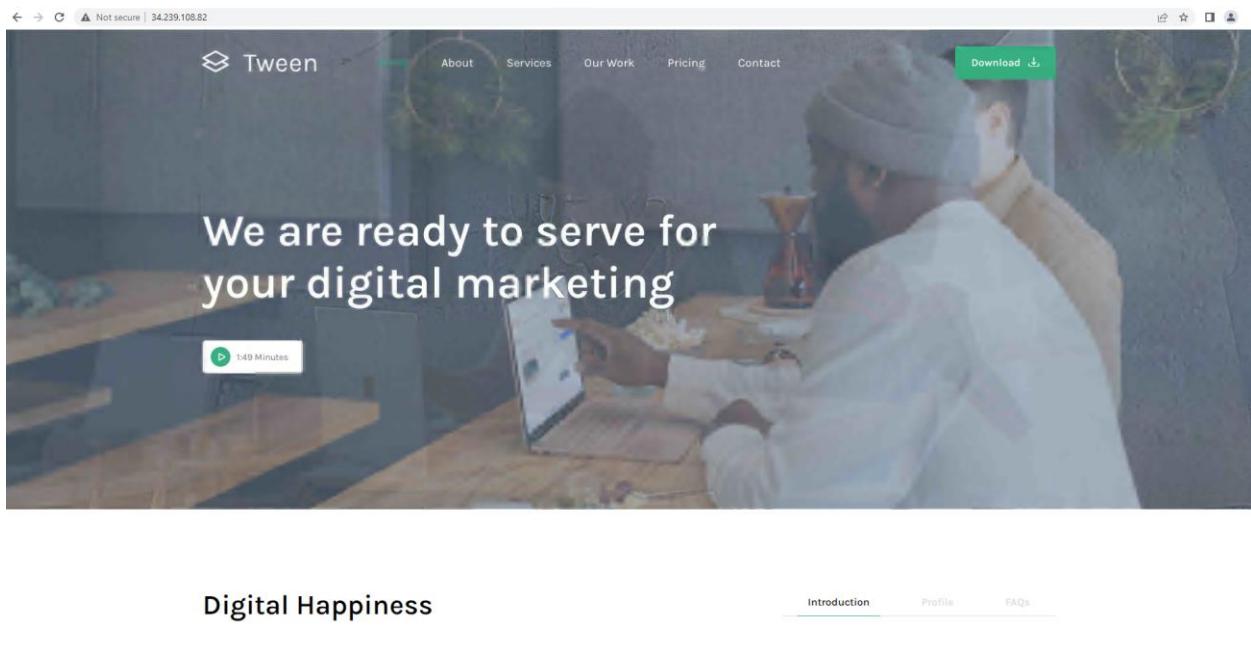


```
root@ip-172-31-86-107:~# ls /var/www/html/images/
dream-about-working-corporation-became-true.jpg
lost+found
maria-stewart-p4tj0g-_aMM-unsplash.jpg
php-programming-html-coding-cyberspace-concept.jpg
services
slide
webpage-browser-digital-icon-symbols-concept.jpg
work
root@ip-172-31-86-107:~#
```

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.

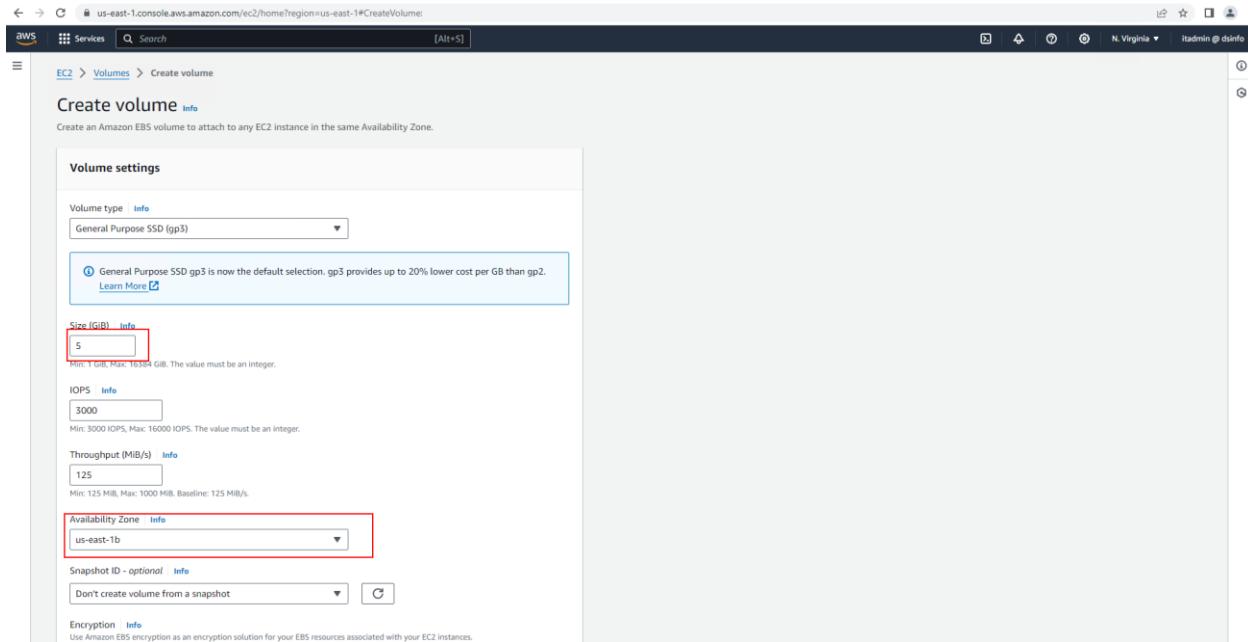
In this section, we will create a new **volume**, attach it to the **EC2 instance**, install a **database**, store the database files on the new volume, and finally, take a **snapshot** of the new volume.

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.

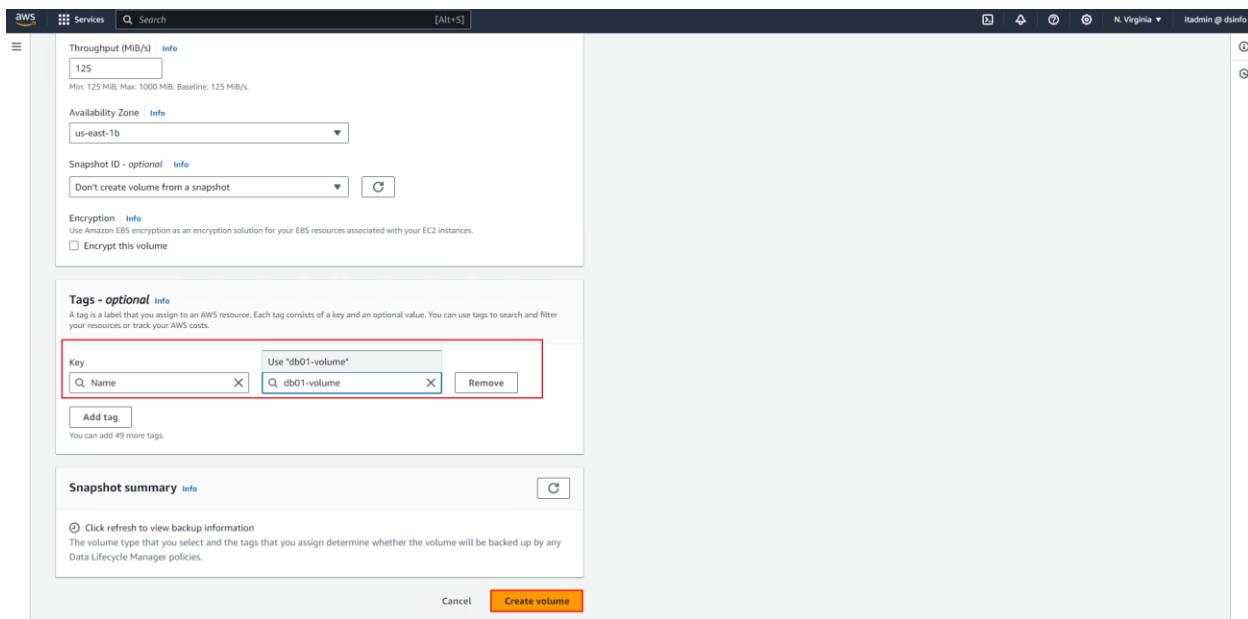
Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot	Created	Availability Zone	Volume state	Alarm stat
web-01-volume	vol-06919be83d30db0a8	gp3	8 GiB	3000	125	snap-04819fa...	2024/03/21 23:16 GMT-7	us-east-1b	In-use	

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



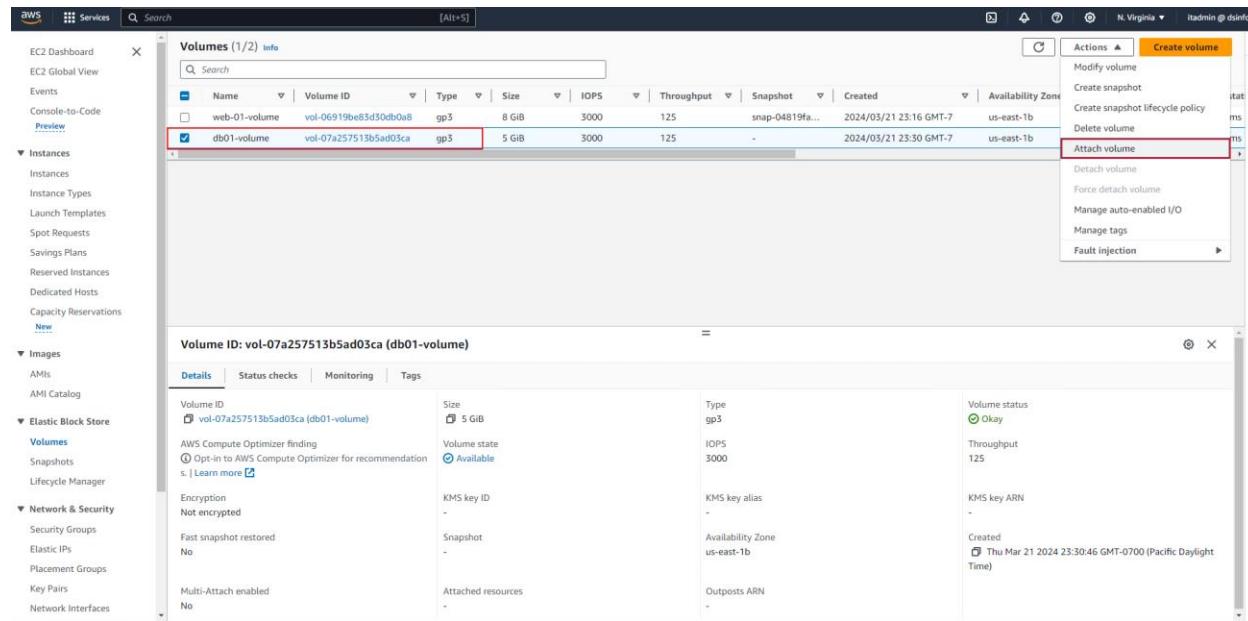
The screenshot shows the 'Create volume' page in the AWS Management Console. Under 'Volume settings', the 'Volume type' is set to 'General Purpose SSD (gp3)'. The 'Size (GiB)' is set to 5. The 'IOPS' is set to 3000. The 'Throughput (MiB/s)' is set to 125. The 'Availability Zone' dropdown is set to 'us-east-1b'. The 'Encryption' section is collapsed. A note at the bottom says 'Create volume'.

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



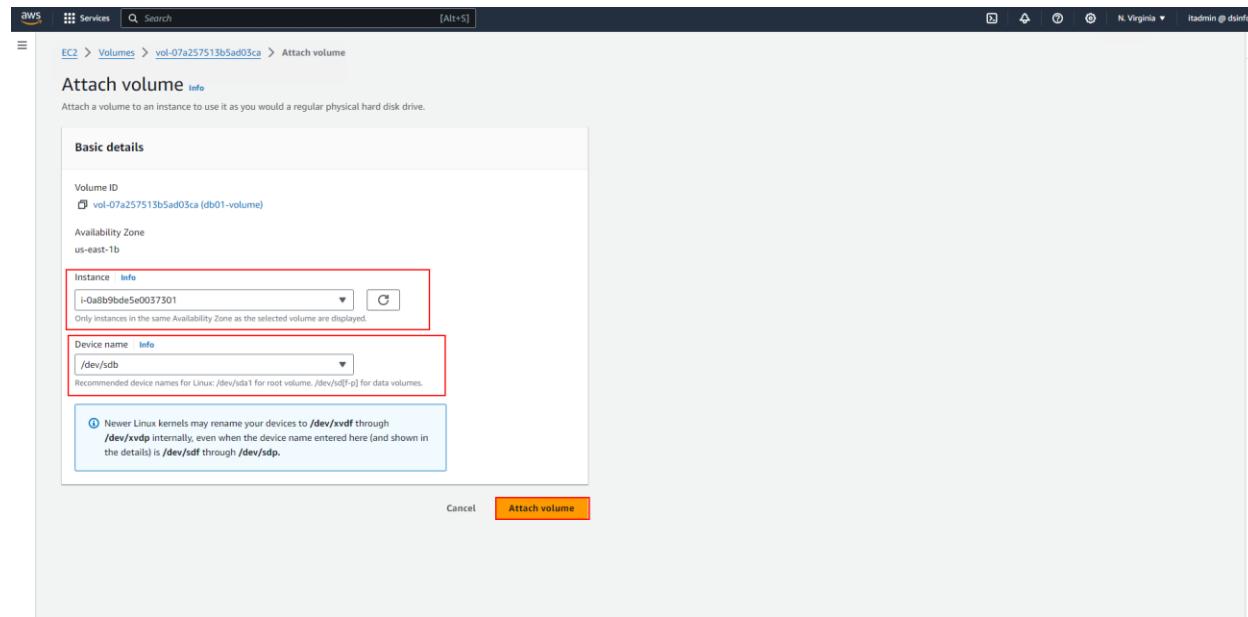
The screenshot shows the 'Create volume' page with the 'Tags - optional' section expanded. A tag named 'db01-volume' is added with the key 'Name'. The 'Create volume' button is highlighted in red.

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



The screenshot shows the AWS EC2 Volumes list. There are two volumes listed: 'web01-volume' and 'db01-volume'. The 'db01-volume' is selected, indicated by a red border around its row. In the top right corner of the list, there is an 'Actions' dropdown menu. The 'Attach volume' option is highlighted with a red border.

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



The screenshot shows the 'Attach volume' dialog box. Under the 'Basic details' section, the 'Volume ID' is set to 'vol-07a257513b5ad03ca (db01-volume)'. The 'Availability Zone' is set to 'us-east-1b'. The 'Instance' dropdown is set to 'i-0a8b9bde5e0037301'. The 'Device name' dropdown is set to '/dev/sdb'. At the bottom of the dialog, there is a note: 'Never Linux kernels may rename your devices to /dev/xvdf through /dev/xvd* internally, even when the device name entered here (and shown in the details) is /dev/sdf through /dev/sdp.' The 'Attach volume' button is highlighted with a red border.

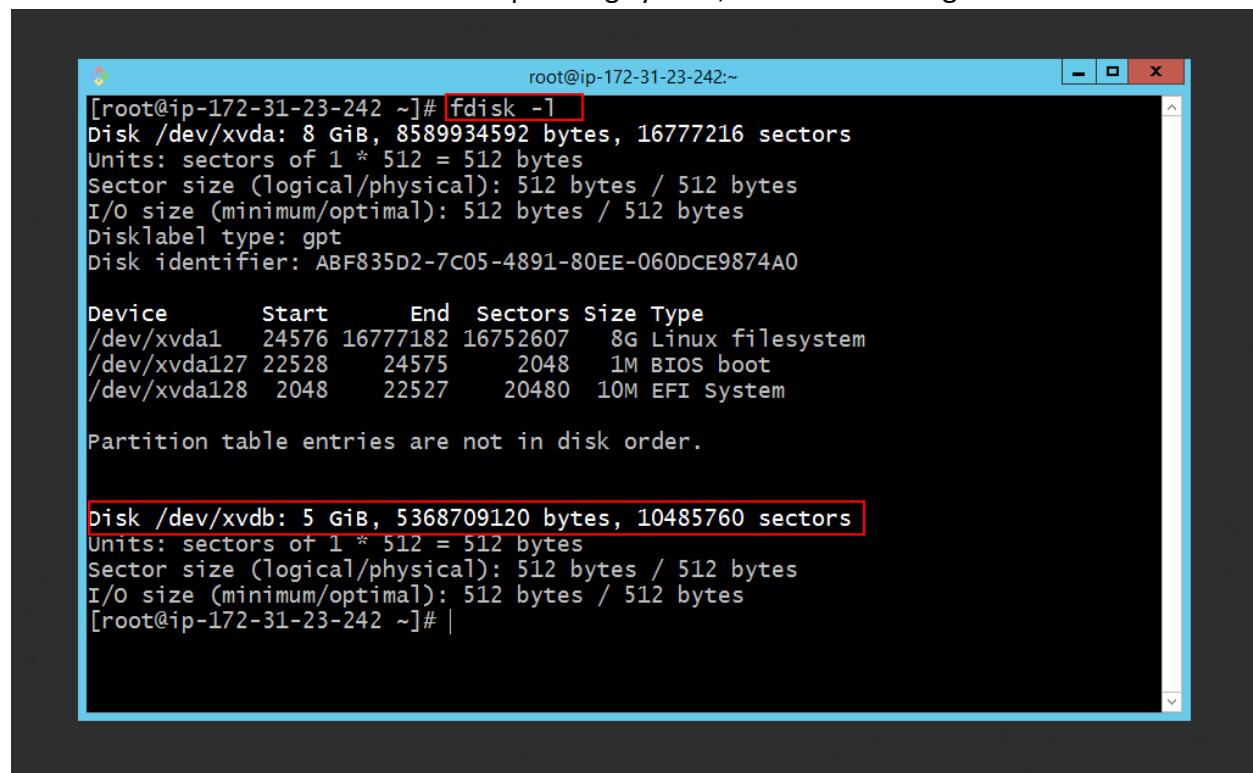
Then, SSH into your **EC2 instance** and switch to the **Root** user environment by using the following command

The screenshot shows a terminal window titled "Administrator@WIN-EN7125AN7JV MINGW64 ~/Downloads". The user runs the command `ssh -i key-pair.pem ec2-user@54.226.181.229`. The system prompts for confirmation about the host's fingerprint, which the user accepts with `yes`. It then adds the host to the list of known hosts. The terminal then displays a stylized logo for Amazon Linux 2023, followed by the URL `https://aws.amazon.com/linux/amazon-linux-2023`. Finally, the user runs `sudo -i`, becoming the root user.

```
Administrator@WIN-EN7125AN7JV MINGW64 ~/Downloads
$ ssh -i key-pair.pem ec2-user@54.226.181.229
The authenticity of host '54.226.181.229 (54.226.181.229)' can't be established.
ED25519 key fingerprint is SHA256:YMWiyDmTbdLBVVA7lITa74/CQTdmFYisQYkwDUGKVPs.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.226.181.229' (ED25519) to the list of known hosts.

      #_
     ~\_\####_
     ~~\####\_
     ~~ \###|
     ~~   \|/_   https://aws.amazon.com/linux/amazon-linux-2023
     ~~    V~'.'->
     ~~~   / \
     ~~..-. / \
     _/m/,' / \
[ec2-user@ip-172-31-23-242 ~]$ sudo -i
[root@ip-172-31-23-242 ~]# |
```

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



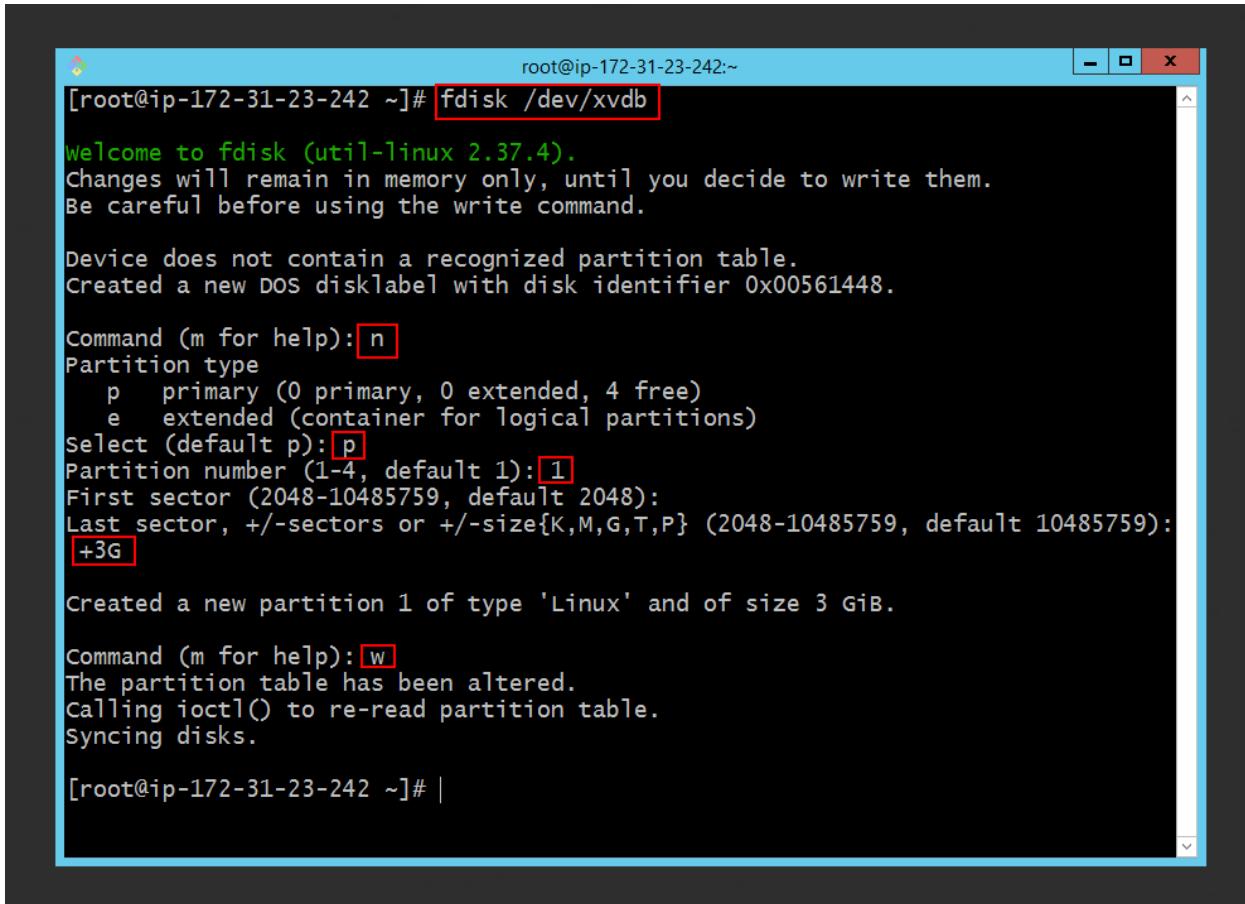
```
root@ip-172-31-23-242:~# fdisk -l
Disk /dev/xvda: 8 GiB, 8589934592 bytes, 16777216 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: ABF835D2-7C05-4891-80EE-060DCE9874A0

Device      Start     End   Sectors Size Type
/dev/xvda1    24576 16777182 16752607   8G Linux filesystem
/dev/xvda127  22528     24575      2048   1M BIOS boot
/dev/xvda128   2048    22527     20480  10M EFI System

Partition table entries are not in disk order.

Disk /dev/xvdb: 5 GiB, 5368709120 bytes, 10485760 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
[root@ip-172-31-23-242:~]# |
```

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.



```
root@ip-172-31-23-242:~# fdisk /dev/xvdb
Welcome to fdisk (util-linux 2.37.4).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x00561448.

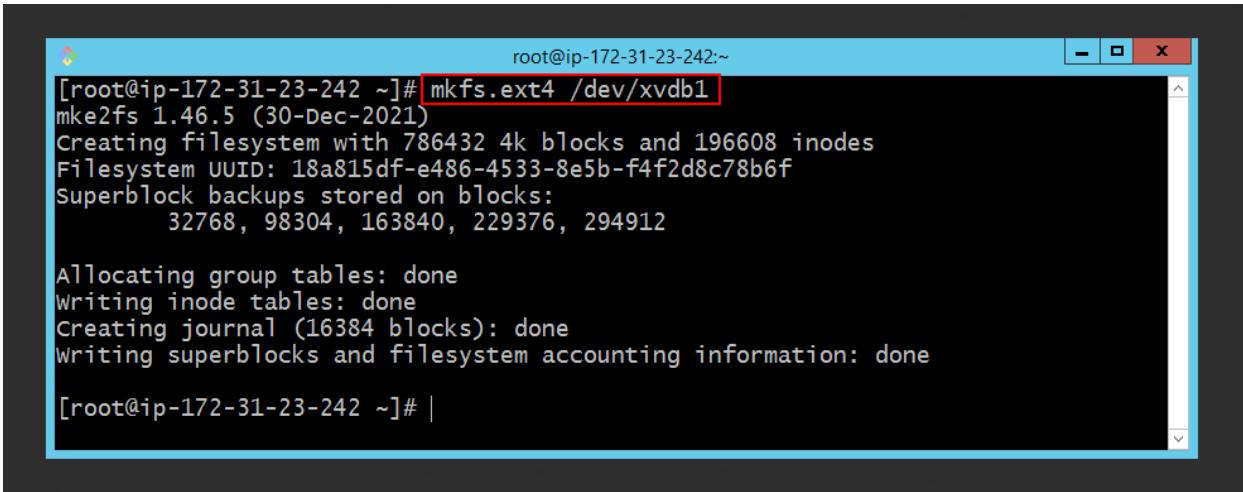
Command (m for help): n
Partition type
  p  primary (0 primary, 0 extended, 4 free)
  e  extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-10485759, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-10485759, default 10485759):
+3G

Created a new partition 1 of type 'Linux' and of size 3 GiB.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

[root@ip-172-31-23-242:~]# |
```

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

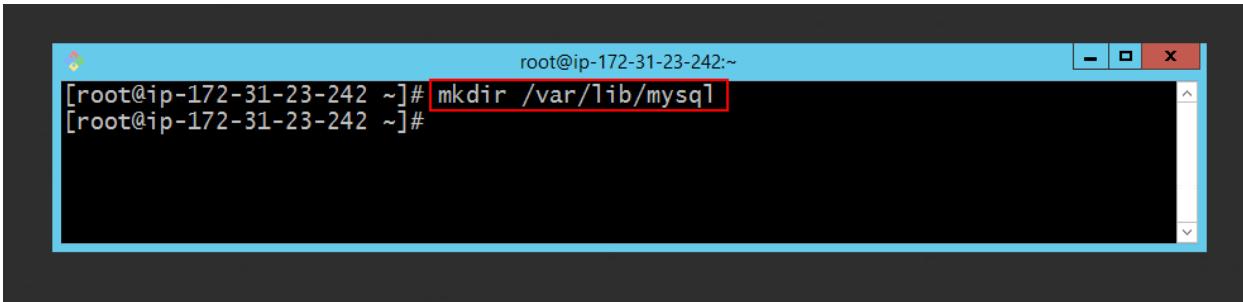


```
root@ip-172-31-23-242 ~]# mkfs.ext4 /dev/xvdb1
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 786432 4k blocks and 196608 inodes
Filesystem UUID: 18a815df-e486-4533-8e5b-f4f2d8c78b6f
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

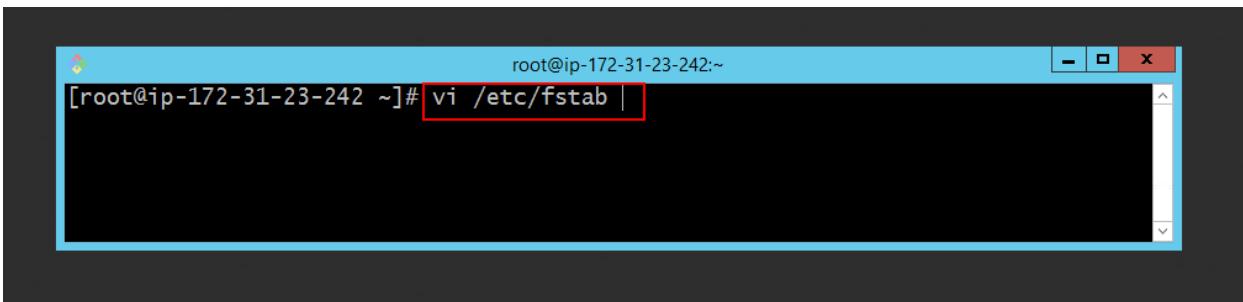
[root@ip-172-31-23-242 ~]# |
```

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



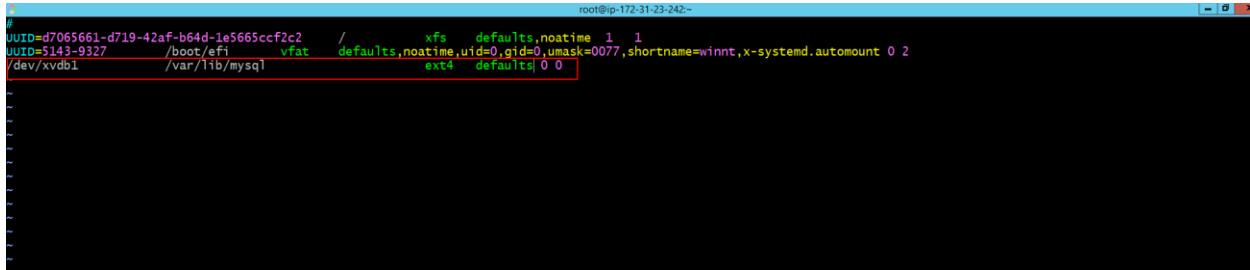
```
root@ip-172-31-23-242 ~]# mkdir /var/lib/mysql
[root@ip-172-31-23-242 ~]#
```

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



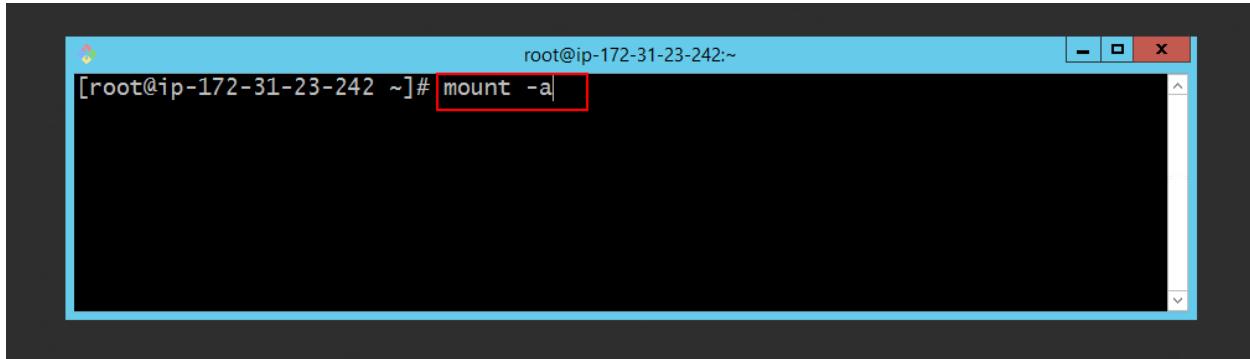
```
root@ip-172-31-23-242 ~]# vi /etc/fstab |
```

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



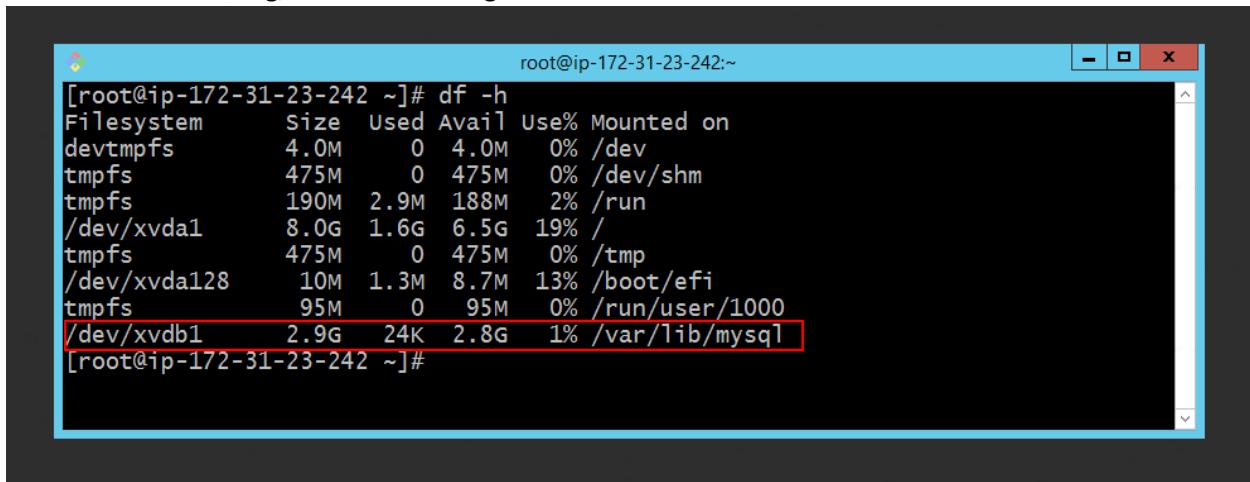
```
#  
UUID=d7065661-d719-42af-b64d-1e5665ccf2c2      /          xfs      defaults,noatime 1 1  
UUID=5143-9327        /boot/efi    vfat      defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount 0 2  
/dev/xvdb1            /var/lib/mysql  ext4      defaults 0 0
```

To mount the partition, use the following command



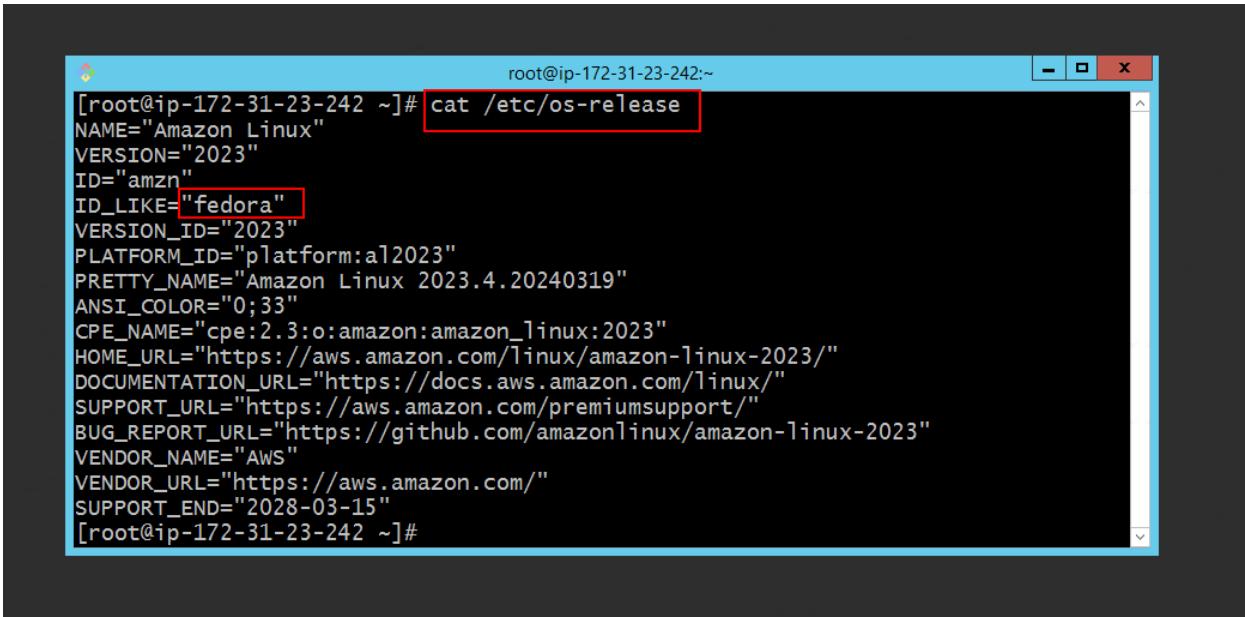
```
[root@ip-172-31-23-242 ~]# mount -a
```

To check the mounting, use the following command



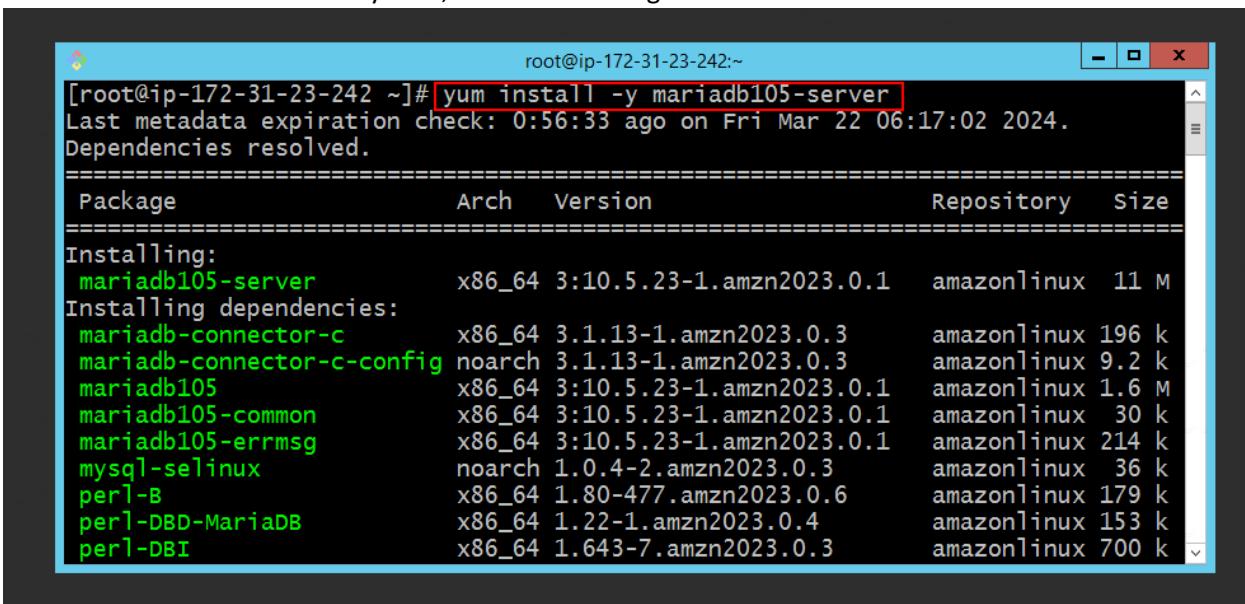
```
[root@ip-172-31-23-242 ~]# df -h  
Filesystem      Size  Used Avail Use% Mounted on  
devtmpfs        4.0M   0    4.0M  0% /dev  
tmpfs          475M   0   475M  0% /dev/shm  
tmpfs          190M  2.9M  188M  2% /run  
/dev/xvda1       8.0G  1.6G  6.5G  19% /  
tmpfs          475M   0   475M  0% /tmp  
/dev/xvda128     10M  1.3M  8.7M  13% /boot/efi  
tmpfs          95M   0   95M  0% /run/user/1000  
[redacted]  
/dev/xvdb1       2.9G  24K  2.8G  1% /var/lib/mysql  
[root@ip-172-31-23-242 ~]#
```

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



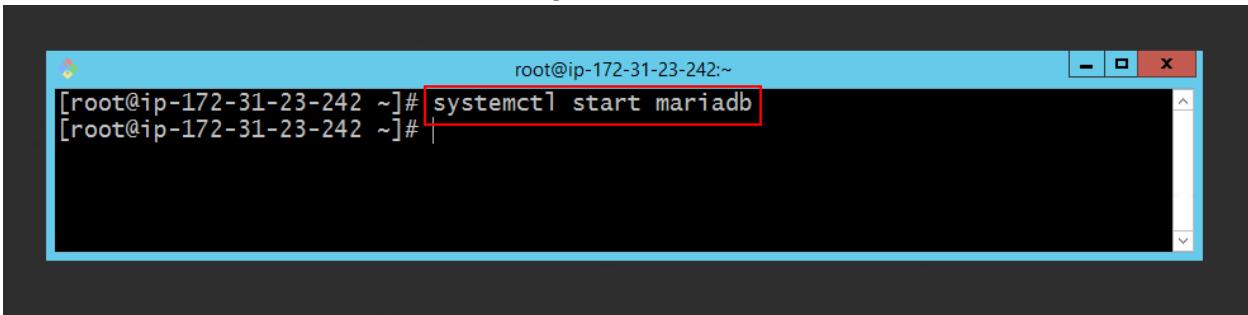
```
root@ip-172-31-23-242 ~]# cat /etc/os-release
NAME="Amazon Linux"
VERSION="2023"
ID="amzn"
ID_LIKE="fedora"
VERSION_ID="2023"
PLATFORM_ID="platform:el2023"
PRETTY_NAME="Amazon Linux 2023.4.20240319"
ANSI_COLOR="0;33"
CPE_NAME="cpe:2.3:o:amazon:amazon_linux:2023"
HOME_URL="https://aws.amazon.com/linux/amazon-linux-2023/"
DOCUMENTATION_URL="https://docs.aws.amazon.com/linux/"
SUPPORT_URL="https://aws.amazon.com/premiumsupport/"
BUG_REPORT_URL="https://github.com/amazonlinux/amazon-linux-2023"
VENDOR_NAME="AWS"
VENDOR_URL="https://aws.amazon.com/"
SUPPORT_END="2028-03-15"
[root@ip-172-31-23-242 ~]#
```

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



```
root@ip-172-31-23-242 ~]# yum install -y mariadb105-server
Last metadata expiration check: 0:56:33 ago on Fri Mar 22 06:17:02 2024.
Dependencies resolved.
=====
 Package           Arch   Version        Repository      Size
=====
Installing:
 mariadb105-server    x86_64 3:10.5.23-1.amzn2023.0.1  amazonlinux 11 M
Installing dependencies:
 mariadb-connector-c  x86_64 3.1.13-1.amzn2023.0.3   amazonlinux 196 k
 mariadb-connector-c-config noarch 3.1.13-1.amzn2023.0.3  amazonlinux 9.2 k
 mariadb105          x86_64 3:10.5.23-1.amzn2023.0.1  amazonlinux 1.6 M
 mariadb105-common   x86_64 3:10.5.23-1.amzn2023.0.1  amazonlinux 30 k
 mariadb105-errmsg   x86_64 3:10.5.23-1.amzn2023.0.1  amazonlinux 214 k
 mysql-selinux       noarch 1.0.4-2.amzn2023.0.3   amazonlinux 36 k
 perl-B              x86_64 1.80-477.amzn2023.0.6   amazonlinux 179 k
 perl-DBD-MariaDB   x86_64 1.22-1.amzn2023.0.4   amazonlinux 153 k
 perl-DBI             x86_64 1.643-7.amzn2023.0.3  amazonlinux 700 k
```

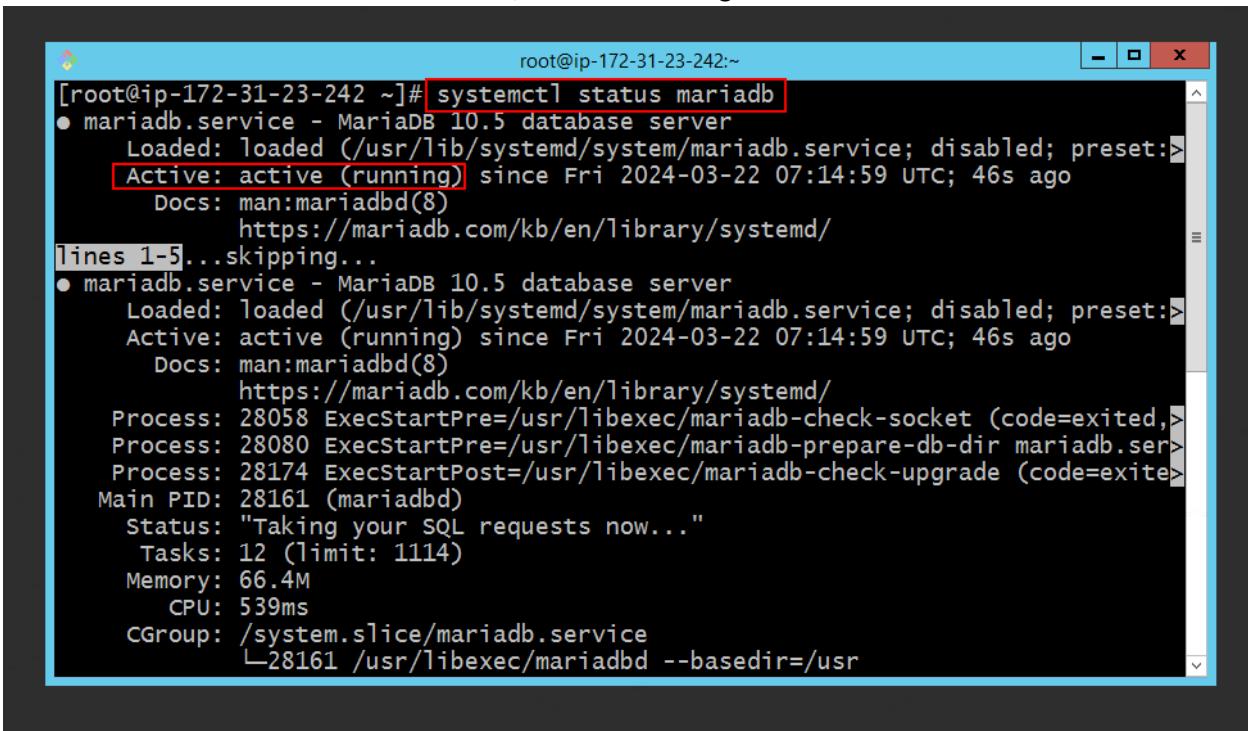
To start the **MariaDB** service, use the following command



```
root@ip-172-31-23-242 ~]# systemctl start mariadb
[root@ip-172-31-23-242 ~]#
```

A screenshot of a terminal window titled 'root@ip-172-31-23-242 ~'. The command 'systemctl start mariadb' is highlighted with a red box. The terminal window has a blue border.

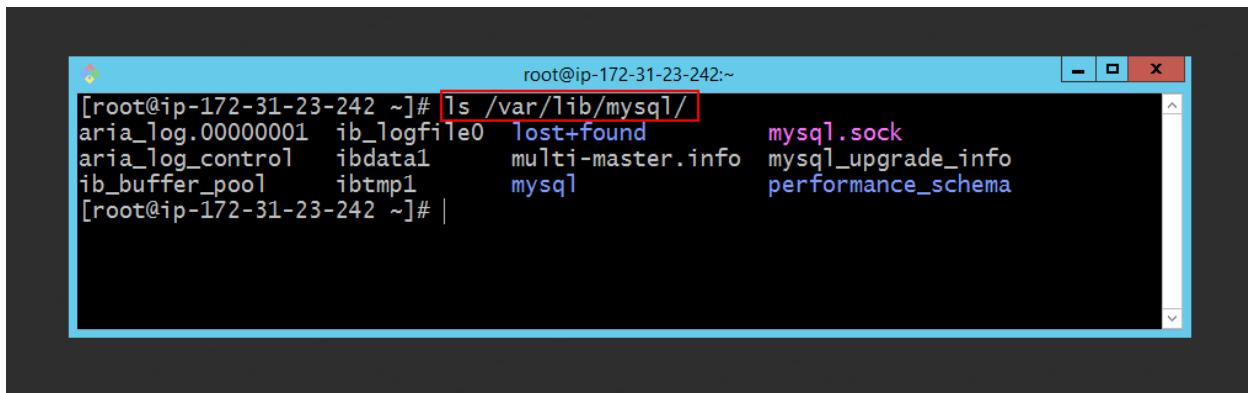
To check the status of the **MariaDB** service, use the following command



```
root@ip-172-31-23-242 ~]# systemctl status mariadb
● mariadb.service - MariaDB 10.5 database server
  Loaded: Loaded (/usr/lib/systemd/system/mariadb.service; disabled; preset:>)
  Active: active (running) since Fri 2024-03-22 07:14:59 UTC; 46s ago
    Docs: man:mariadb(8)
           https://mariadb.com/kb/en/library/systemd/
lines 1-5...skipping...
● mariadb.service - MariaDB 10.5 database server
  Loaded: Loaded (/usr/lib/systemd/system/mariadb.service; disabled; preset:>)
  Active: active (running) since Fri 2024-03-22 07:14:59 UTC; 46s ago
    Docs: man:mariadb(8)
           https://mariadb.com/kb/en/library/systemd/
  Process: 28058 ExecStartPre=/usr/libexec/mariadb-check-socket (code=exited,>
  Process: 28080 ExecStartPre=/usr/libexec/mariadb-prepare-db-dir mariadb.ser>
  Process: 28174 ExecStartPost=/usr/libexec/mariadb-check-upgrade (code=exite>
Main PID: 28161 (mariadb)
  Status: "Taking your SQL requests now..."
  Tasks: 12 (limit: 1114)
 Memory: 66.4M
    CPU: 539ms
  CGroup: /system.slice/mariadb.service
          └─28161 /usr/libexec/mariadb --basedir=/usr
```

A screenshot of a terminal window titled 'root@ip-172-31-23-242 ~'. The command 'systemctl status mariadb' is highlighted with a red box. The terminal window has a blue border.

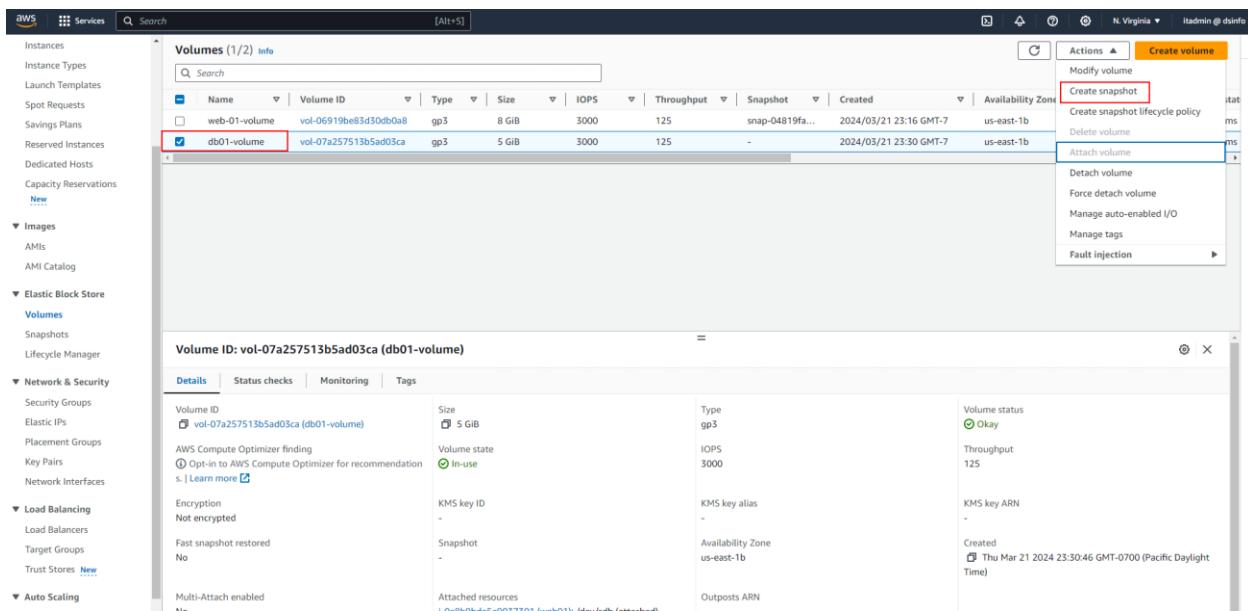
As shown in the image below, after installing the database, the database files are located in the path mounted on the new volume.



```
root@ip-172-31-23-242 ~]# ls /var/lib/mysql/
aria_log.00000001  ib_logfile0  lost+found      mysql.sock
aria_log_control   ibdata1     multi-master.info mysql_upgrade_info
ib_buffer_pool    ibtmp1     mysql
[root@ip-172-31-23-242 ~]#
```

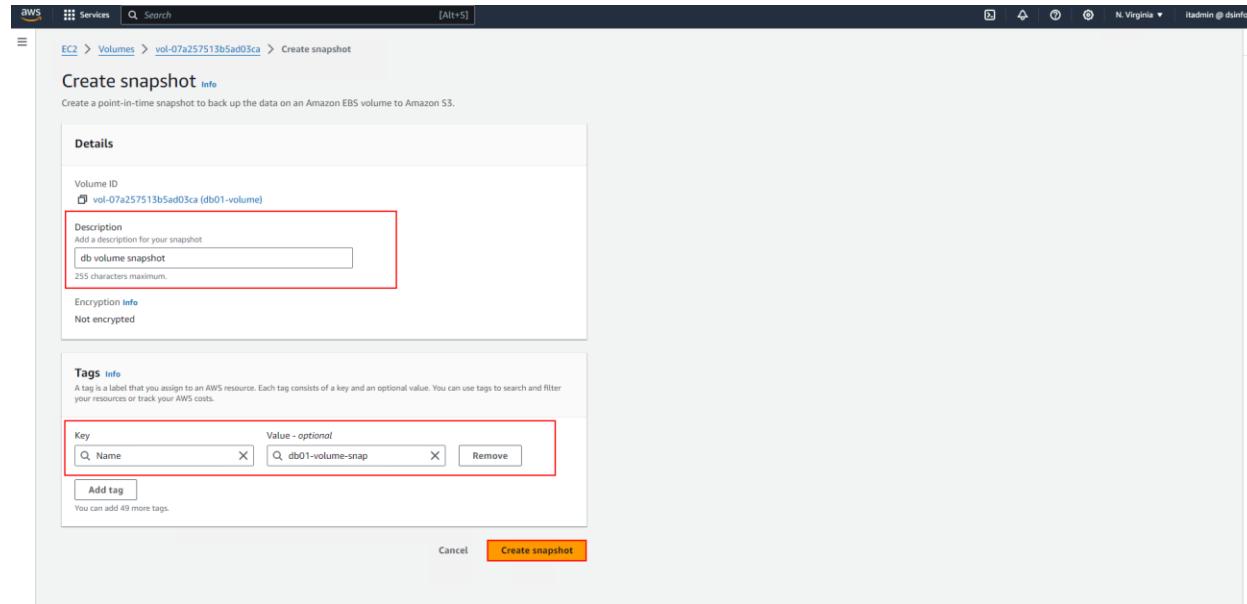
How to Take a Snapshot of a Volume

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

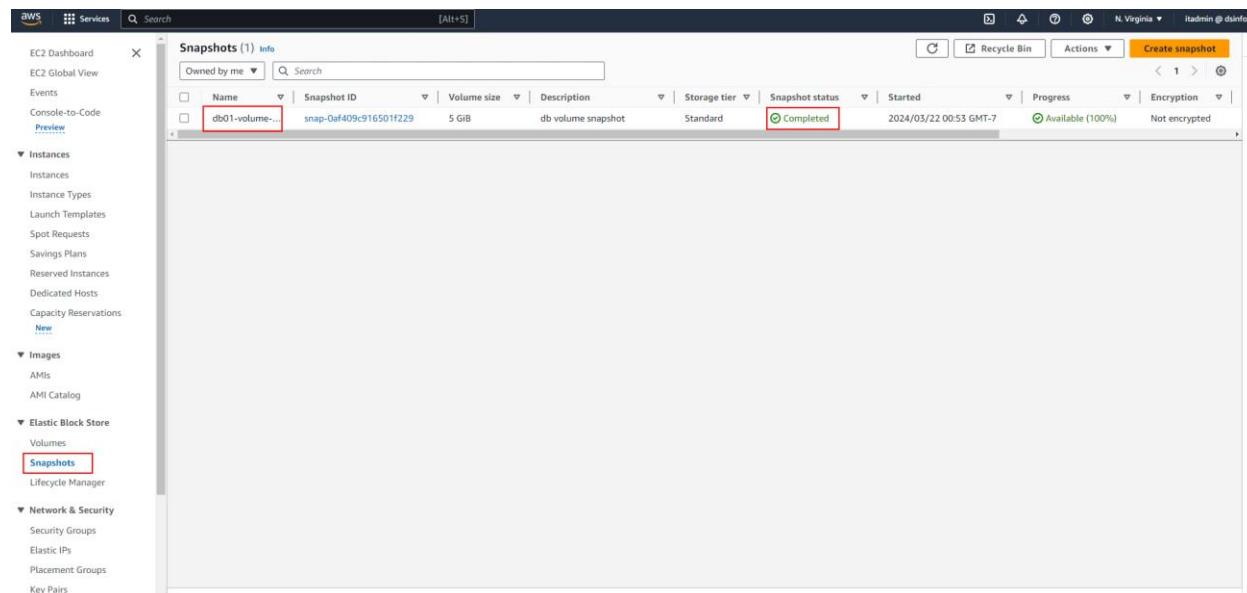


The screenshot shows the AWS Management Console with the Volumes page open. On the left, there's a navigation sidebar with various services like Instances, Images, and Auto Scaling. The main area displays a table of volumes. One volume, 'db01-volume' with Volume ID 'vol-07a257513b5ad03ca', is selected and highlighted with a red box. In the Actions menu (which is also highlighted with a red box), the 'Create snapshot' option is selected. The 'Details' tab is active in the volume preview pane at the bottom.

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

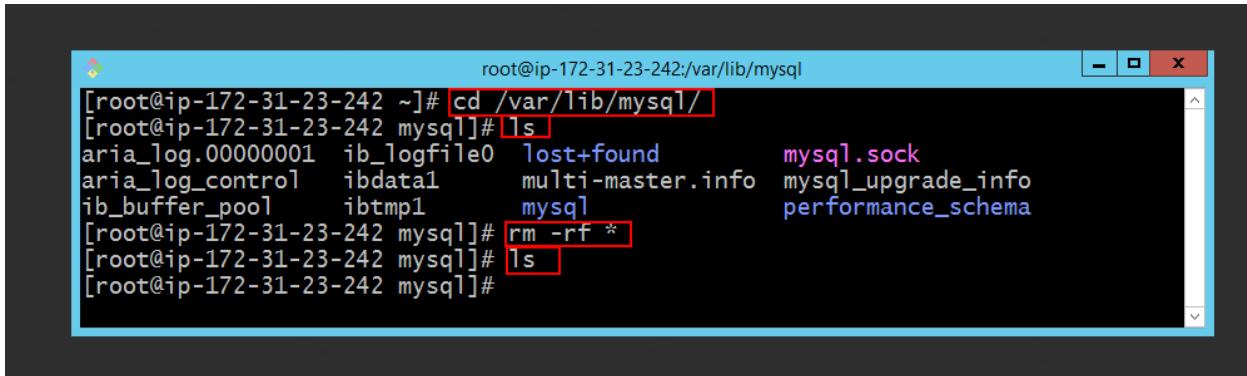


To view the created snapshot, click on the **Snapshots** section.



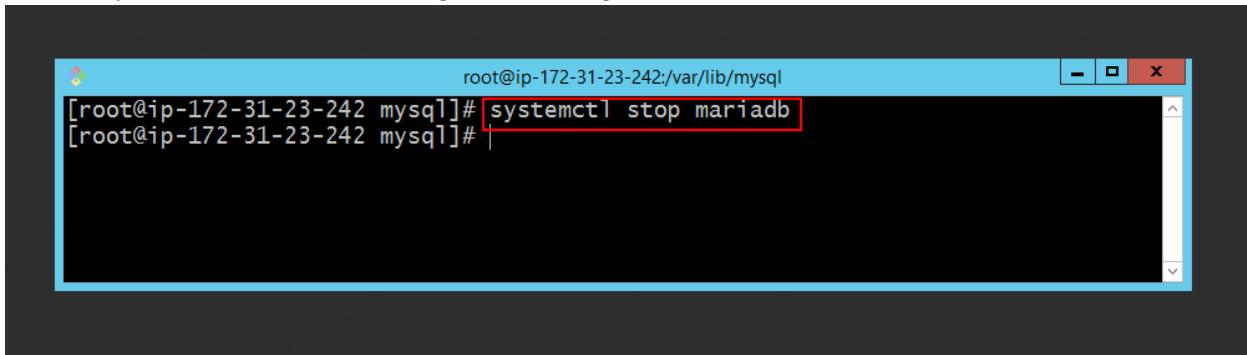
How to Use a Snapshot

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



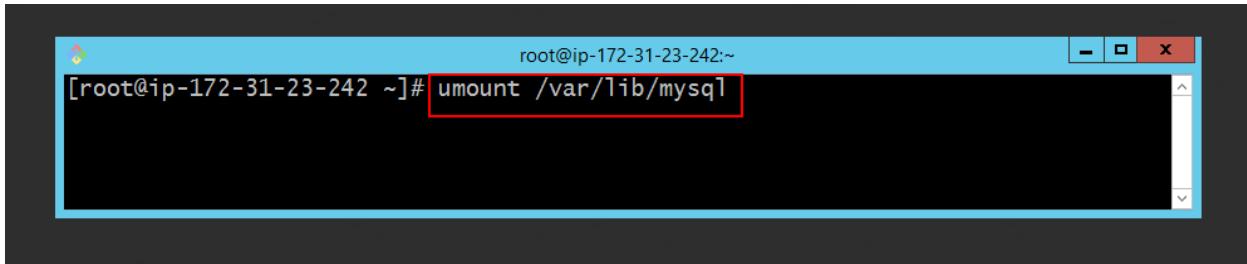
```
root@ip-172-31-23-242 ~]# cd /var/lib/mysql/
[root@ip-172-31-23-242 mysql]# ls
aria_log.00000001 ib_logfile0 lost+found      mysql.sock
aria_log_control ibdata1    multi-master.info  mysql_upgrade_info
ib_buffer_pool ibtmp1    mysql                performance_schema
[root@ip-172-31-23-242 mysql]# rm -rf *
[root@ip-172-31-23-242 mysql]# ls
[root@ip-172-31-23-242 mysql]#
```

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



```
root@ip-172-31-23-242:/var/lib/mysql#
[root@ip-172-31-23-242 mysql]# systemctl stop mariadb
[root@ip-172-31-23-242 mysql]# |
```

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



```
root@ip-172-31-23-242:~#
[root@ip-172-31-23-242 ~]# umount /var/lib/mysql
```

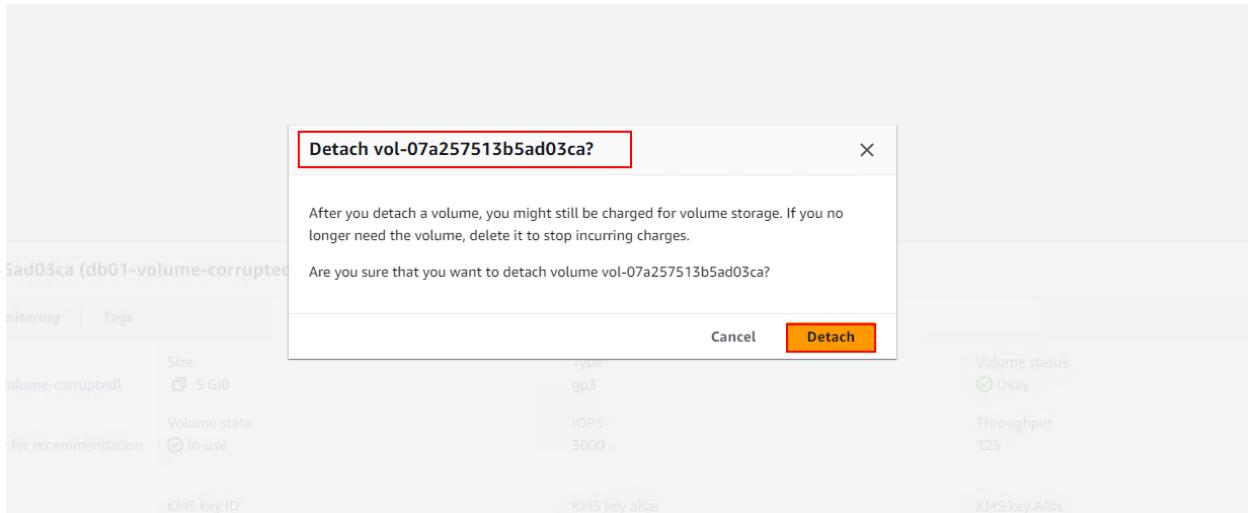
Next, click on the **Volumes** section, and then name the new volume.

The screenshot shows the AWS EC2 Volumes page. On the left, the navigation menu includes 'EC2 Dashboard', 'EC2 Global View', 'Events', 'Console-to-Code', 'Instances', 'Images', 'Elastic Block Store' (with 'Volumes' selected), and 'Network & Security'. The main content area displays a table of volumes. One volume, 'db01-volume', is selected and has its name changed to 'db01-volume-corrupted'. A modal dialog titled 'Volume ID: vol-07a257513b5ad03ca (db01-volume)' shows the details of the volume, including its ID, type (gp3), size (5 GiB), and creation date (Mar 21, 2024). The 'Actions' menu at the top right of the modal includes options like 'Create volume', 'Modify volume', 'Create snapshot', 'Delete volume', 'Attach volume', and 'Detach volume'.

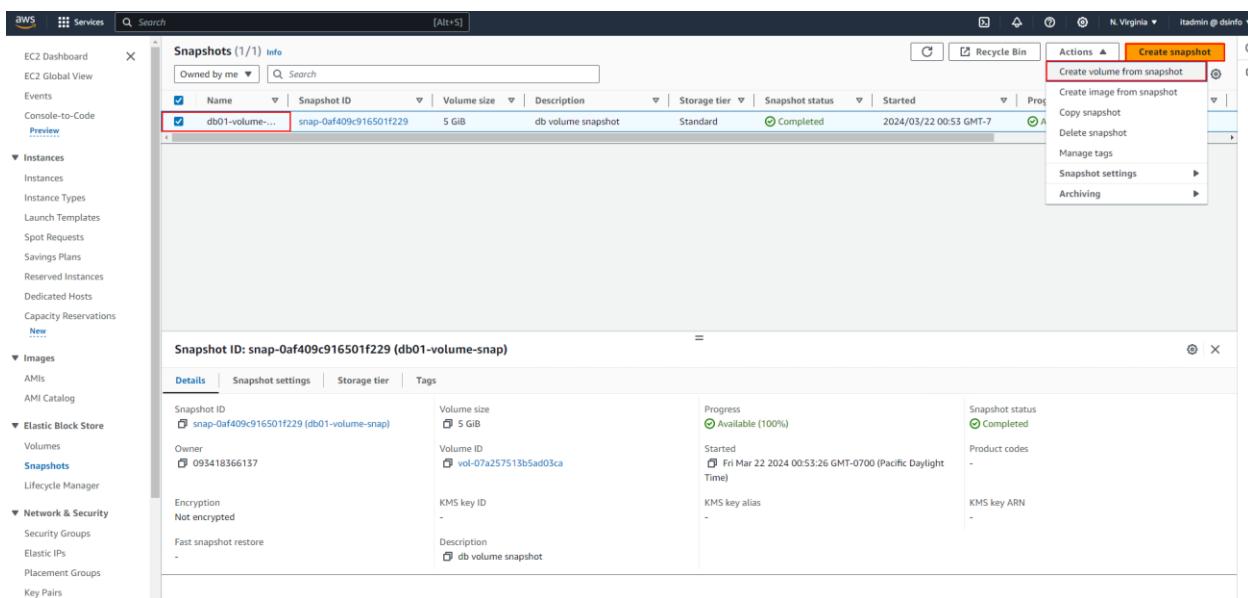
At this stage, select the **new volume**, then from the **Actions** menu, click on **Detach Volume**.

This screenshot is similar to the previous one but shows the 'db01-volume-corrupted' volume selected. The 'Actions' menu is open, and the 'Detach volume' option is highlighted with a red box. The rest of the interface and volume details are identical to the previous screenshot.

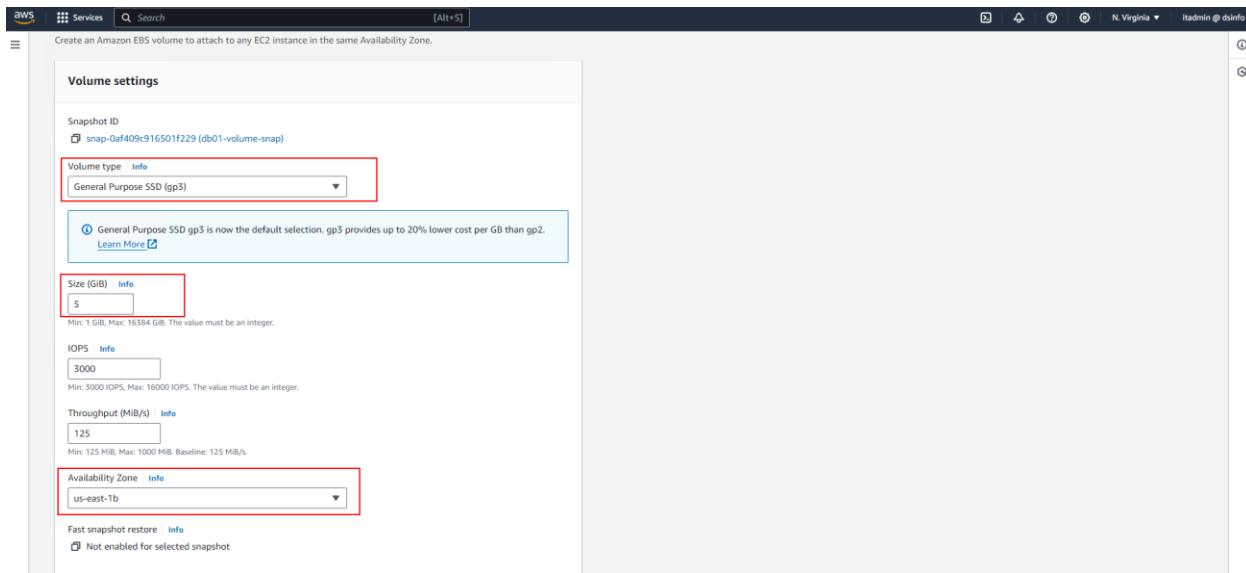
At this stage, click the **Detach** button to disconnect the volume from the EC2 instance.



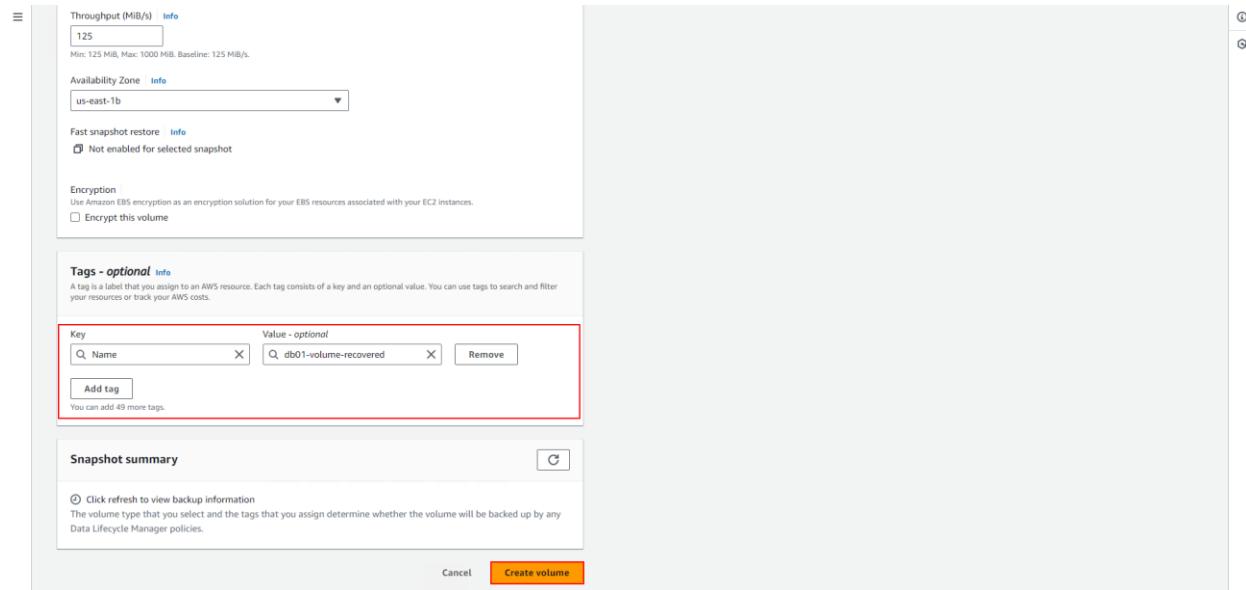
Next, in the **Snapshots** section, click on the snapshot that we took in the previous steps, and from the **Actions** menu, click on **Create Volume from Snapshot**.



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.

Volumes (1/3) Info

Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot	Created	Actions
web-01-volume	vol-06919be83d50db0a8	gp3	8 GiB	3000	125	snap-04819fa...	2024/03/21 23:16 GMT-7	Edit
db01-volume-corrupted	vol-07a257513b5ad03ca	gp3	5 GiB	3000	125	-	2024/03/21 23:30 GMT-7	Edit
db01-volume-recovered	vol-071342c8f6763bd39	gp3	5 GiB	3000	125	snap-0af409c...	2024/03/22 01:09 GMT-7	Edit

Volume ID: vol-071342c8f6763bd39 (db01-volume-recovered)

Details Status checks Monitoring Tags

Volume ID vol-071342c8f6763bd39 (db01-volume-recovered)	Size 5 GiB	Type gp3	Volume status OK
AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations Learn more	Volume state Available	IOPS 3000	Throughput 125
Encryption Not encrypted	KMS key ID -	KMS key alias -	KMS key ARN -
Fast snapshot restored No	Snapshot snap-0af409c916501f229	Availability Zone us-east-1b	Created Fri Mar 22 2024 01:09:56 GMT-0700 (Pacific Daylight Time)
Multi-Attach enabled No	Attached resources -	Outposts ARN -	

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

EC2 > Volumes > vol-071342c8f6763bd39 > Attach volume

Attach volume [Info](#)

Attach a volume to an instance to use it as you would a regular physical hard disk drive.

Basic details

Volume ID
[vol-071342c8f6763bd39 \(db01-volume-recovered\)](#)

Availability Zone
us-east-1b

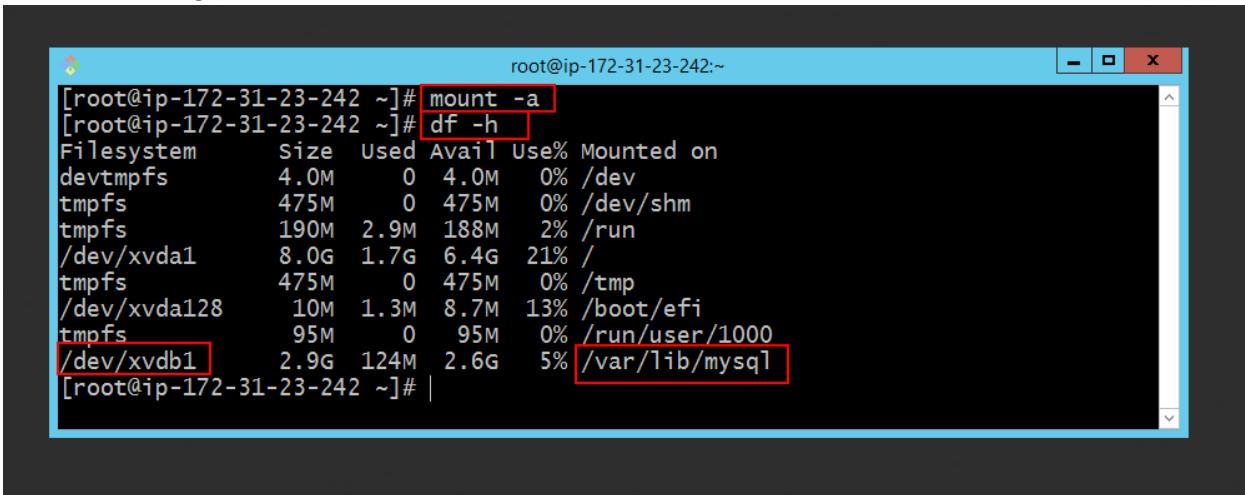
Instance [Info](#)
 [Cancel](#)
Only instances in the same Availability Zone as the selected volume are displayed.

Device name [Info](#)
 [Cancel](#)
Recommended device names for Linux: /dev/sda1 for root volume, /dev/sdf-p1 for data volumes.

Never Linux kernels may rename your devices to /dev/xvdf through /dev/xvdः internally, even when the device name entered here (and shown in the details) is /dev/sdf through /dev/sdः.

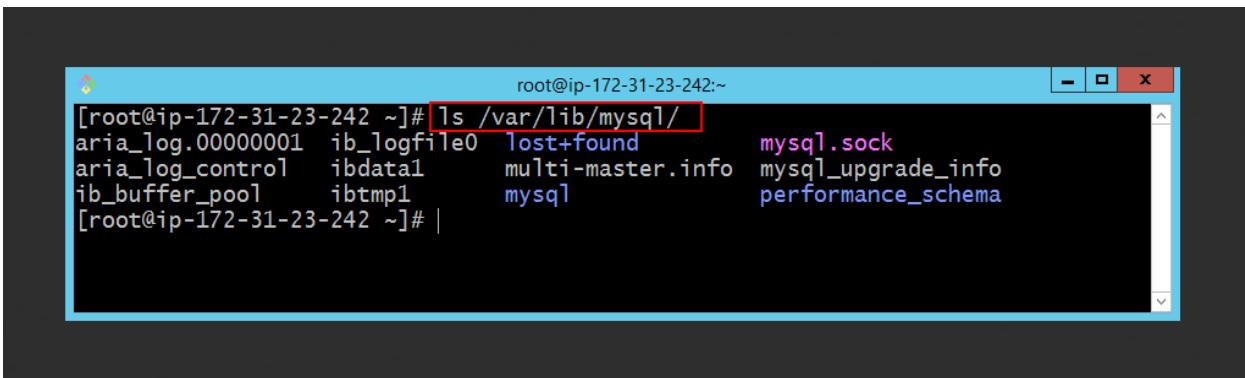
Cancel [Attach volume](#)

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



```
root@ip-172-31-23-242 ~]# mount -a
[root@ip-172-31-23-242 ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/devtmpfs        4.0M   0    4.0M  0% /dev
tmpfs           475M   0   475M  0% /dev/shm
tmpfs           190M  2.9M  188M  2% /run
/dev/xvda1       8.0G  1.7G  6.4G  21% /
tmpfs           475M   0   475M  0% /tmp
/dev/xvda128     10M  1.3M  8.7M  13% /boot/efi
tmpfs            95M   0   95M  0% /run/user/1000
/dev/xvdb1       2.9G  124M  2.6G  5% /var/lib/mysql
[root@ip-172-31-23-242 ~]# |
```

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



```
root@ip-172-31-23-242 ~]# ls /var/lib/mysql/
aria_log.00000001  ib_logfile0  Lost+found          mysql.sock
aria_log_control   ibdata1     multi-master.info   mysql_upgrade_info
ib_buffer_pool     ibtmp1     mysql               performance_schema
[root@ip-172-31-23-242 ~]# |
```

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.

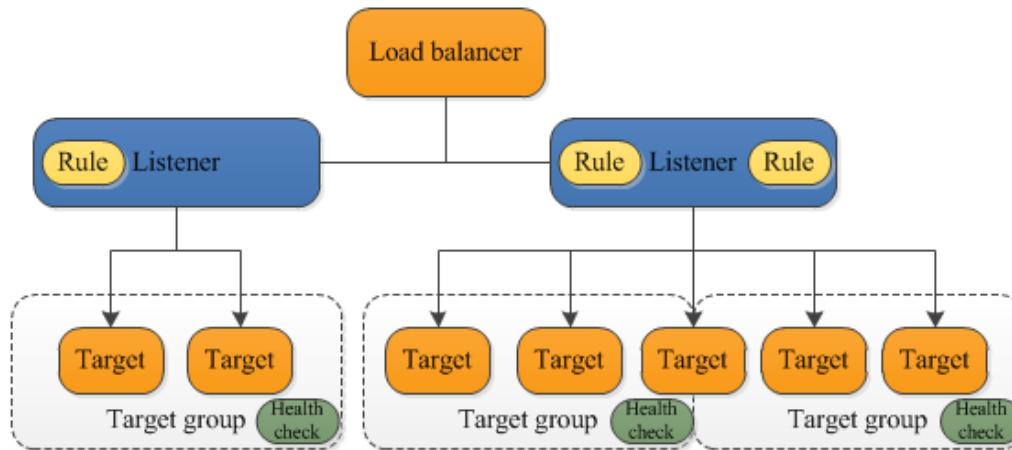
Types of AWS ELB

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

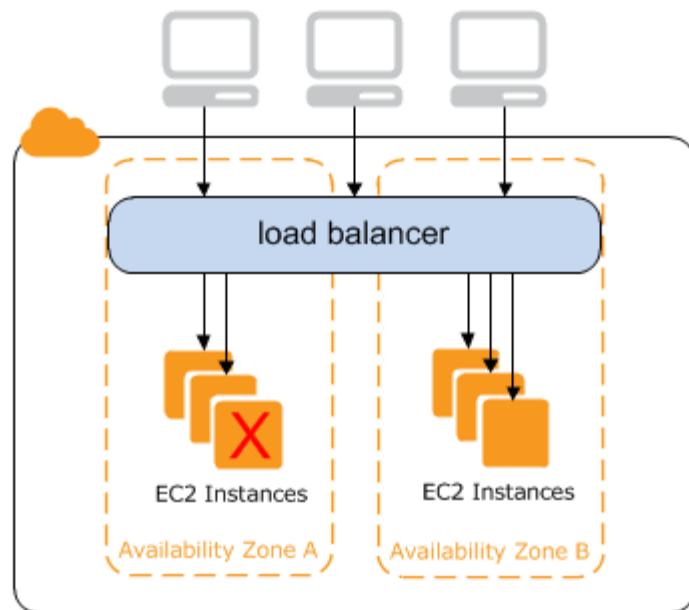
This **Load Balancer** provides **high performance** and is **very costly**.

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.



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**.

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with options like EC2 Dashboard, EC2 Global View, Events, and Instances (which is selected and highlighted with a red box). Under Instances, there are sub-options: Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, and Key Pairs. The main content area shows a table titled 'Instances Info' with one row: 'Instance: i-0a8b9bde5e0037301 (web01)'. The 'Details' tab is selected. The instance details include: Instance ID: i-0a8b9bde5e0037301 (web01); Public IPv4 address: -; Instance state: Terminated; Instance type: t2.micro; Auto-assigned IP address: -; Private IPv4 addresses: -; Public IPv4 DNS: -; Elastic IP addresses: -; VPC ID: -; AWS Compute Optimizer finding: -.

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

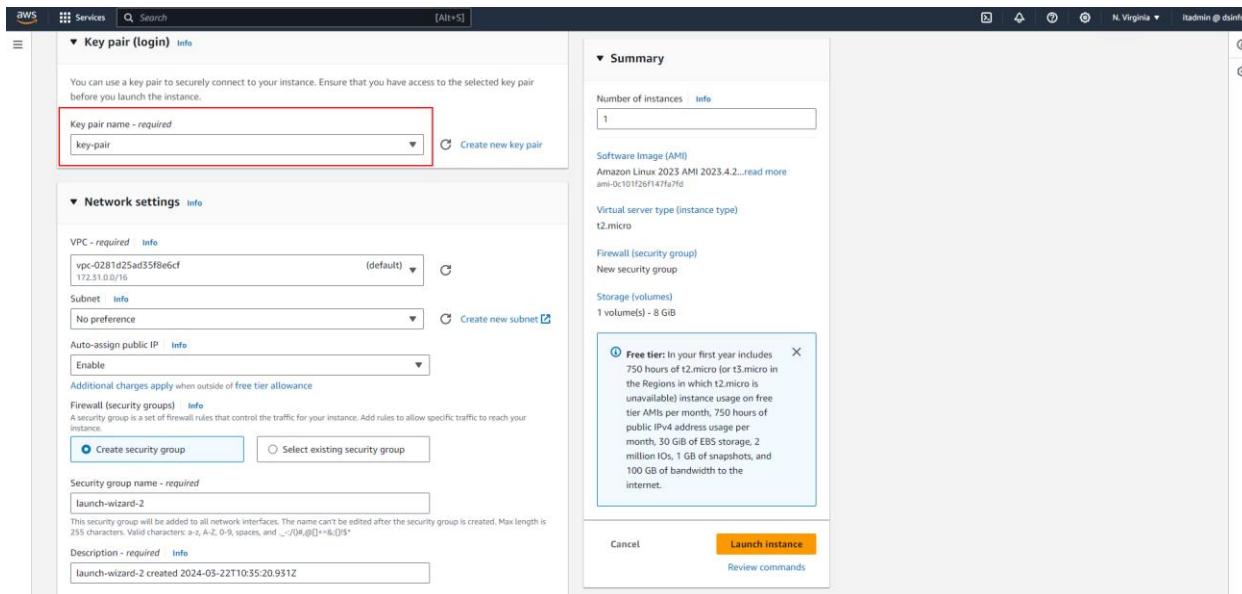
The screenshot shows the 'Launch an instance' wizard. Step 1: 'Name and tags' (Name: web01). Step 2: 'Application and OS Images (Amazon Machine Image)' (Search bar: 'Search our full catalog including 1000s of application and OS images'). Step 3: 'Quick Start' (Amazon Linux (AMI) selected, highlighted with a red box). Step 4: 'Summary' (Number of instances: 1; Software Image (AMI): Amazon Linux 2023 AMI 2023.4.2...; Virtual server type (instance type): t2.micro; Firewall (security group): New security group; Storage (volumes): 1 volume(s) - 8 GiB). A callout box for the free tier states: 'Free tier: In your first year includes 750 hours of t2.micro (or t3.micro is unavailable) instance usage on tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.' Step 5: 'Launch instance' button.

141

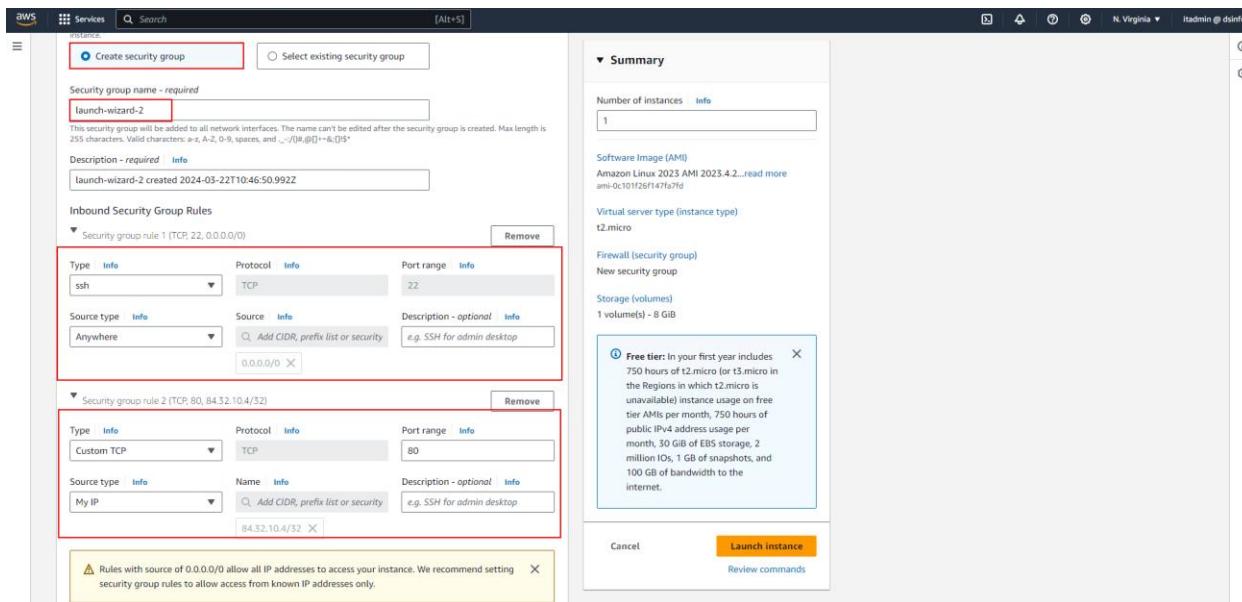
Instructor: Fariborz Fallahzadeh

Email Address:fariborz.fallahzadeh@gmail.com

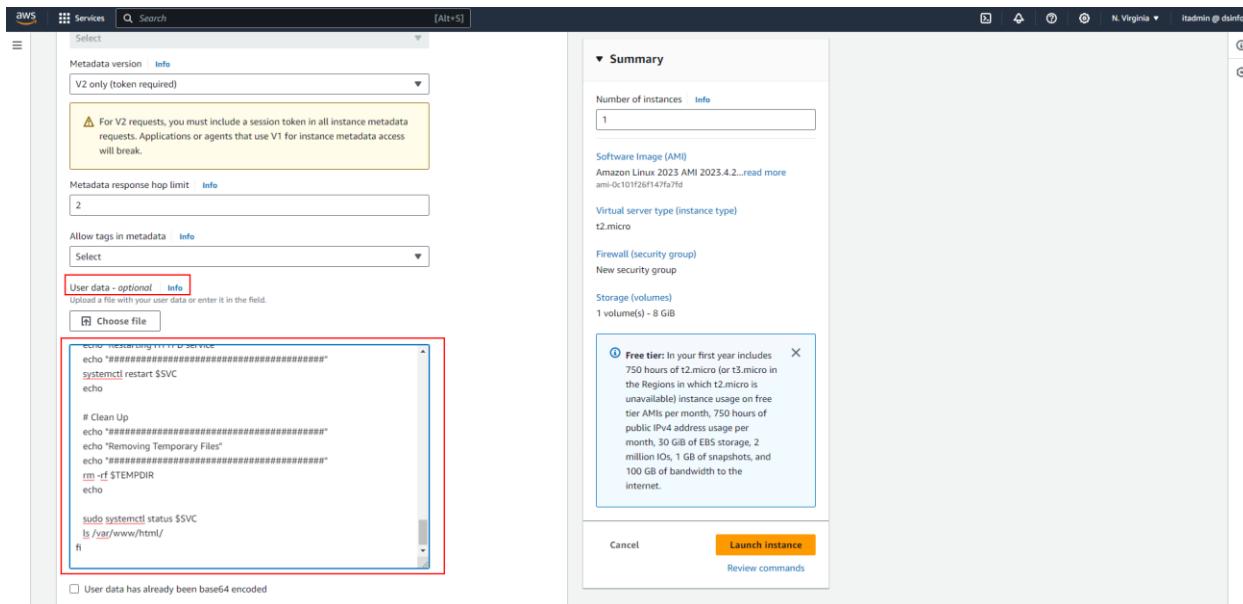
At this stage, you need to define a **Key Pair**.



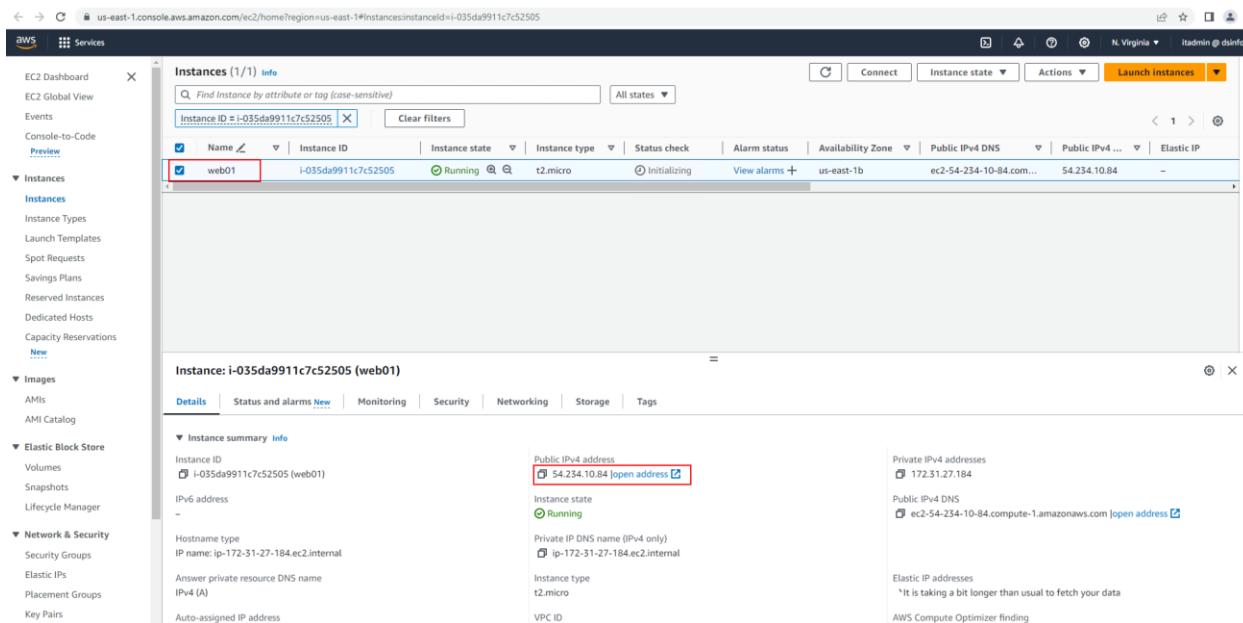
At this stage, you need to configure the **Security Group** settings for this EC2 instance. Open ports **80** and **22** in the **Inbound Rules** section.



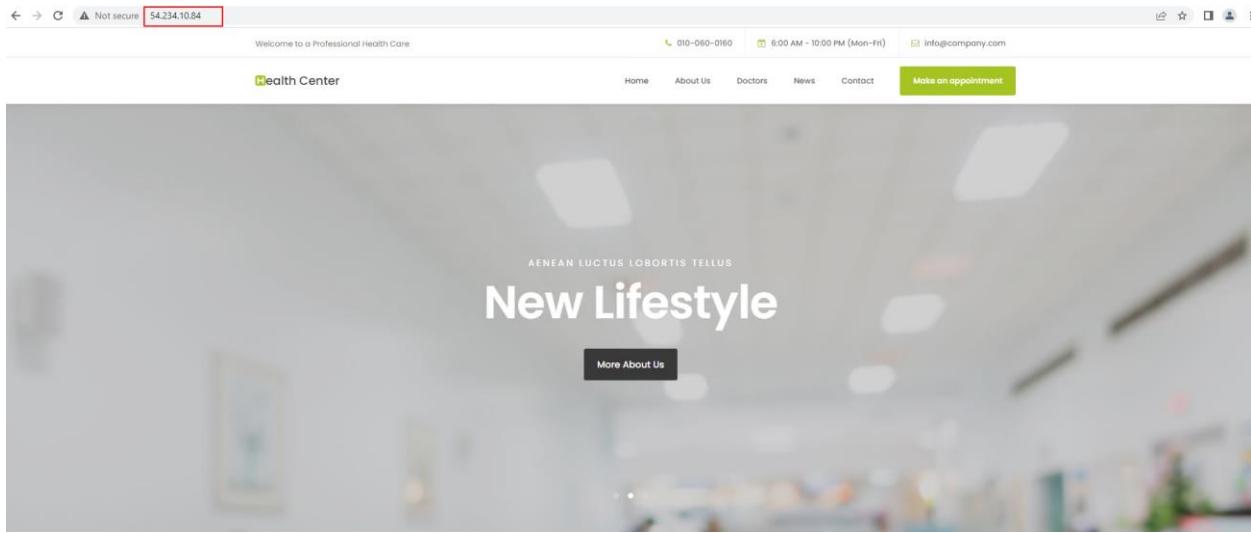
At this stage, paste the **web server setup script** into the **User Data** section.



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**.

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like EC2 Dashboard, Instances (selected), Images, and Network & Security. The main area displays a table of instances. One instance, 'web01' (Instance ID: i-035da9911c7c52505), is selected and highlighted with a red box. In the 'Actions' dropdown menu, the 'Image and templates' option is also highlighted with a red box. Below the table, there's a detailed view for the selected instance, showing its configuration and network details.

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

The screenshot shows the 'Create Image' dialog box. At the top, there is a field for 'Image name' containing 'Health-AM'. Below it is a 'Image description - optional' field with the placeholder 'Image description'. Underneath are settings for 'No reboot' (unchecked) and 'Instance volumes'. A table lists a single volume: 'Storage type' (EBS), 'Device' (/dev/sda1), 'Size' (8), 'Volume type' (EBS General Purpose S...), 'IOPS' (3000), 'Throughput' (1000), 'Delete on termination' (checked), and 'Encrypted' (checked). Below the table is a note: 'During the image creation process, Amazon EC2 creates a snapshot of each of the above volumes.' Under 'Tags - optional', there are two radio button options: 'Tag image and snapshots together' (selected) and 'Tag image and snapshots separately'. Both options include the note 'Tag the image and the snapshots with the same tag.' or 'Tag the image and the snapshots with different tags.'. At the bottom, there is a 'Cancel' button and a prominent orange 'Create image' button.

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

The screenshot shows the 'Launch Templates' page in the AWS Management Console. The left sidebar has 'Launch Templates' selected under the 'Instances' section. The main area displays a table with columns: 'Launch Template ID', 'Launch Template Name', 'Default Version', 'Latest Version', 'Create Time', and 'Created By'. A search bar at the top is empty. An orange 'Create launch template' button is located in the top right corner of the main area. Below the table, a message says 'Loading Launch Templates...'.

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

The screenshot shows the 'Create launch template' wizard. In the 'Launch template name and description' section, the 'Launch template name - required' field contains 'Health-Template' and the 'Template version description' field contains 'V1'. A note below says 'Must be unique to this account. Max 128 chars. No spaces or special characters like %, ^, @.' Below these fields are sections for 'Auto Scaling guidance' and 'Template tags'. In the 'Launch template contents' section, there is a note: 'Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.' At the bottom right of this section is a link to 'Application and OS Images (Amazon Machine Image)'. On the right side of the screen, a 'Summary' panel lists 'Software Image (AMI)', 'Virtual server type (instance type)', 'Firewall (security group)', and 'Storage (volumes)'. A callout box highlights the 'Free tier' information: 'In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet.'

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

The screenshot shows the 'Amazon Machine Image (AMI)' selection interface. It features a search bar at the top with the placeholder 'Search our full catalog including 1000s of application and OS images'. Below the search bar are three tabs: 'Recents', 'My AMIs' (which is selected and highlighted with a red box), and 'Quick Start'. Underneath the tabs are three filter buttons: 'Don't include in launch template', 'Owned by me' (which is selected and highlighted with a red box), and 'Shared with me'. The main area displays a list of AMIs under the heading 'Amazon Machine Image (AMI)'. One item is highlighted with a red box: 'Health-AMI' (ami-056e19828de61afec). Below this item, detailed information is shown: '2024-03-22T11:03:20.000Z', 'Virtualization: hvm', 'ENI enabled: true', and 'Root device type: ebs'. To the right of this list is a 'Browse more AMIs' link. Below the AMI list are sections for 'Description', 'Architecture' (x86_64), and 'AMI ID' (ami-056e19828de61afec). At the bottom of the interface are sections for 'Instance type' (with a dropdown menu showing 'Don't include in launch template') and 'Advanced' settings, including 'All generations' and 'Compare instance types' buttons. On the right side of the screen, a 'Summary' panel lists 'Software Image (AMI)', 'Virtual server type (instance type)', 'Firewall (security group)', and 'Storage (volumes)'. A callout box highlights the 'Free tier' information: 'In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet.'

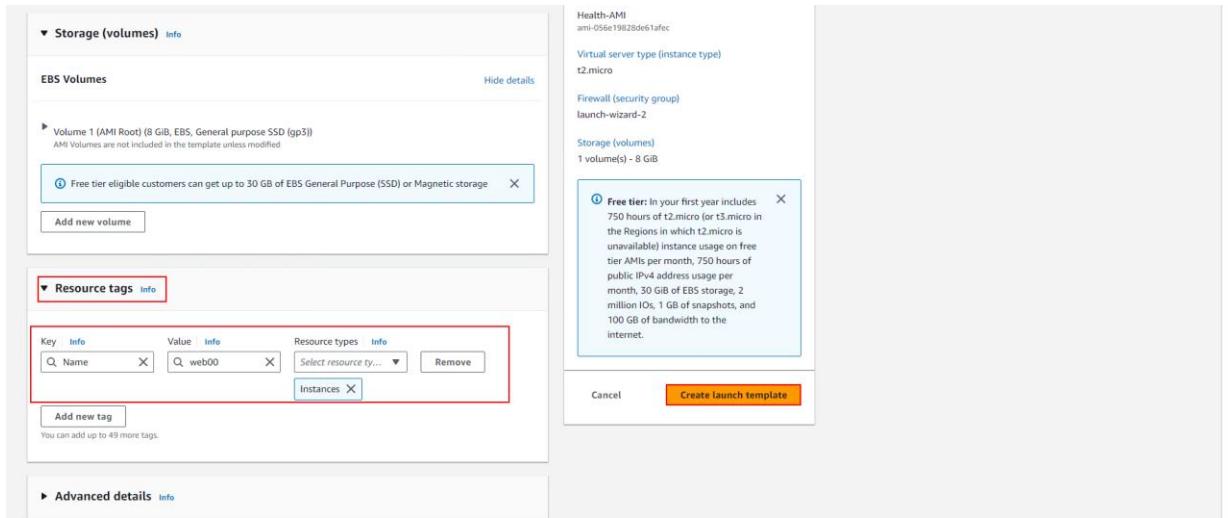
At this stage, you need to specify the **Instance Type** and the **Key Pair**.

The screenshot shows the AWS Launch Wizard configuration interface. The 'Instance type' section is highlighted with a red box, showing 't2.micro' selected. The 'Key pair (login)' section is also highlighted with a red box, showing 'key-pair' selected. To the right, a detailed description of the 'Free tier' is provided, stating it includes 750 hours of t2.micro instance usage per month, 750 hours of public IPv4 address usage per month, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. A large orange 'Create launch template' button is at the bottom right.

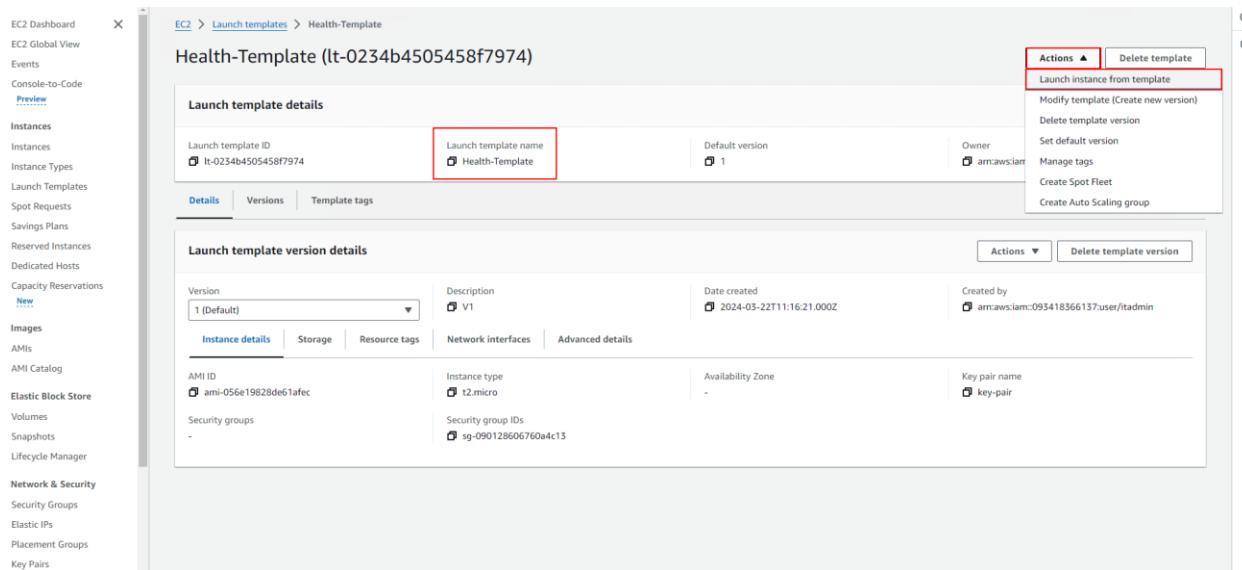
At this stage, you need to configure the **Security Group**.

The screenshot shows the AWS Launch Wizard configuration interface. The 'Network settings' section is highlighted with a red box, showing 'Select existing security group' selected. Below it, the 'Security groups info' section shows 'launch-wizard-2 sg-090128606760a4c13' selected. To the right, a detailed description of the 'Free tier' is provided, stating it includes 750 hours of t2.micro instance usage per month, 750 hours of public IPv4 address usage per month, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. A large orange 'Create launch template' button is at the bottom right.

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**.

The screenshot shows the 'Launch instance from template' step in the AWS EC2 console. On the left, under 'Choose a launch template', a dropdown menu labeled 'Source template' is open, showing 'Health-Template' selected. This dropdown is highlighted with a red box. Below it, another dropdown shows '1 (Default) V1'. To the right, the 'Summary' section displays the following details:

- Number of instances: 1
- Software Image (AMI): Health-AMI ami-056e19828de61afec
- Virtual server type (instance type): t2.micro
- Firewall (security group): launch-wizard-2
- Storage (volumes): 1 volume(s) - 8 GiB

A tooltip for the AMI selection area states: 'Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.'

At the bottom, there are 'Cancel', 'Launch instance' (which is orange), and 'Review commands' buttons.

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

The screenshot shows the continuation of the 'Launch instance from template' process. On the left, under 'Resource tags', a new tag is being added with the key 'Name' and the value 'web002'. The 'Value' field is highlighted with a red box. The right panel shows the same summary details as the previous screenshot, including the 'Free tier' tooltip for the AMI selection.

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

The screenshot shows the AWS EC2 Instances page. The left sidebar includes options like EC2 Dashboard, EC2 Global View, Events, Console-to-Code, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, and AMI Catalog. The main content area displays a table titled 'Instances (2) Info' with columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, Public IPv4 IP, and Elastic IP. Two instances are listed: 'web001' (Instance ID: i-035da9911c7c52505, State: Running, Type: t2.micro, Status: 2/2 checks passed, Zone: us-east-1b, DNS: ec2-54-234-10-84.com..., IP: 54.234.10.84) and 'web002' (Instance ID: i-00be1e861e41bc292, State: Running, Type: t2.micro, Status: 2/2 checks passed, Zone: us-east-1b, DNS: ec2-34-201-104-7.com..., IP: 34.201.104.7). Both instances have their names highlighted with red boxes.

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

The screenshot shows the AWS EC2 Target groups page. The left sidebar includes options like Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces, Load Balancing (selected), Load Balancers, Target Groups (selected), and Trust Stores. The main content area shows a table titled 'Target groups Info' with columns: Name, ARN, Port, Protocol, Target type, Load balancer, and VPC ID. A message states 'No target groups' and 'You don't have any target groups in us-east-1'. A 'Create target group' button is visible. The 'Target Groups' section below shows '0 target groups selected' and the message 'Select a target group above.' The 'Target Groups' link in the sidebar is highlighted with a red box.

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

The screenshot shows the 'Specify group details' step of the 'Create target group' wizard. It includes sections for 'Basic configuration', 'Health checks', and 'Attributes'. The 'Instances' target type is selected, and the target group name is set to 'Health-TG'. The 'HTTP' health check protocol and path '/' are also specified. The 'Next' button is visible at the bottom right.

EC2 > Target groups > Create target group

Step 1
Specify group details

Step 2
Register targets

Specify group details

Your load balancer routes requests to the targets in a target group and performs health checks on the targets.

Basic configuration

Settings in this section can't be changed after the target group is created.

Choose a target type

Instances

- Supports load balancing to instances within a specific VPC.
- Facilitates the use of [Amazon EC2 Auto Scaling](#) to manage and scale your EC2 capacity.

IP addresses

- Supports load balancing to VPC and on-premises resources.
- Facilitates routing to multiple IP addresses and network interfaces on the same instance.
- Offers flexibility with microservice based architectures, simplifying inter-application communication.
- Supports IPv6 targets, enabling end-to-end IPv6 communication, and IPv4-to-IPv6 NAT.

Lambda function

- Facilitates routing to a single Lambda function.
- Accessible to Application Load Balancers only.

Application Load Balancer

- Offers the flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC.
- Facilitates using static IP addresses and PrivateLink with an Application Load Balancer.

Target group name

Health-TG

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

HTTP/2

HTTP2 Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.

gRPC Send requests to targets using gRPC. Supported when the request protocol is gRPC.

Health checks

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

Health check protocol

HTTP

Health check path

/

Up to 1024 characters allowed.

Advanced health check settings

Attributes

Certain default attributes will be applied to your target group. You can view and edit them after creating the target group.

Tags - optional

Consider adding tags to your target group. Tags enable you to categorize your AWS resources so you can more easily manage them.

Cancel **Next**

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

EC2 > Target groups > Create target group

Step 1
Specify group details

Step 2
Register targets

Register targets

This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (2/2)

Instance ID	Name	State	Security groups	Zone	Private IPv4 address
i-00be1e861e41bc292	web002	Running	launch-wizard-2	us-east-1b	172.31.22.92
i-035da9911c7c52505	web001	Running	launch-wizard-2	us-east-1b	172.31.27.184

2 selected

Ports for the selected instances
Ports for routing traffic to the selected instances.
80
1-65535 (separate multiple ports with commas)

Include as pending below

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

EC2 > Target groups > Create target group

Step 1
Specify group details

Step 2
Register targets

Review targets

Targets (2)

Instance ID	Name	Port	State	Security groups	Zone	Private IPv4 address	Subnet ID	Launch time
i-00be1e861e41bc292	web002	80	Running	launch-wizard-2	us-east-1b	172.31.22.92	subnet-02e0a980e794cbc1f	March 22, 2024, 04:19 (UTC-07:00)
i-035da9911c7c52505	web001	80	Running	launch-wizard-2	us-east-1b	172.31.27.184	subnet-02e0a980e794cbc1f	March 22, 2024, 03:51 (UTC-07:00)

2 pending

Cancel Previous **Create target group**

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

The screenshot shows the AWS EC2 Target Groups page. On the left sidebar, under the 'Load Balancing' section, the 'Target Groups' option is selected and highlighted with a red box. In the main content area, the 'Target groups (1/1) Info' table has one row for 'Health-TG'. The 'Targets' tab is selected. Under 'Registered targets (2) Info', there are two entries: 'web002' and 'web001'. Both targets have a green status icon indicating they are 'Normal'. A red box highlights the 'Targets' tab and the 'Normal' status for both targets.

Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
<input checked="" type="checkbox"/> Health-TG	arn:aws:elasticloadbalanc...	80	HTTP	Instance	None associated	vpc-0281d25ad35f8e6cf

Instance ID	Name	Port	Zone	Health status	Health status details	Launch time	Anomaly detection...
<input type="checkbox"/> i-00be1e861e41bc292	web002	80	us-east-1b	○ Unused	Target group is not co...	March 22, 2024, 04:19 (UTC-07:00)	Normal
<input type="checkbox"/> i-035da9911c7c52505	web001	80	us-east-1b	○ Unused	Target group is not co...	March 22, 2024, 03:51 (UTC-07:00)	Normal

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

The screenshot shows the AWS EC2 Load Balancers page. On the left sidebar, under the 'Load Balancing' section, the 'Load Balancers' option is selected and highlighted with a red box. In the main content area, the 'Load balancers' table shows 'No load balancers'. Below the table, a message says 'You don't have any load balancers in us-east-1'. A large red box highlights the 'Create load balancer' button at the bottom of the page.

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
------	----------	-------	--------	--------------------	------	--------------

0 load balancers selected
Select a load balancer above.

Create load balancer

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

Compare and select load balancer type

A complete feature-by-feature comparison along with detailed highlights is also available. Learn more [Learn more](#)

Load balancer types		
Application Load Balancer Info	Network Load Balancer Info	Gateway Load Balancer Info
Choose an Application Load Balancer when you need a flexible feature set for your applications with HTTP or HTTPS traffic. Operating at the request level, Application Load Balancers provide advanced routing and visibility features targeted at application architectures, including microservices and containers.	Choose a Network Load Balancer when you need ultra-high performance, TLS offloading at scale, centralized certificate deployment, support for UDP, and static IP addresses for your applications. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second securely while maintaining ultra-low latencies.	Choose a Gateway Load Balancer when you need to deploy and manage a fleet of third-party virtual appliances that support GENEVE. These appliances enable you to improve security, compliance, and policy controls.
Create	Create	Create

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

EC2 > Load balancers > Create Application Load Balancer

Create Application Load Balancer [Info](#)

The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and containers, based on request attributes. When the load balancer receives a connection request, it evaluates the listener rules in priority order to determine which rule to apply, and if applicable, it selects a target from the target group for the rule action.

► How Application Load Balancers work

Basic configuration

Load balancer name
Name must be unique within your AWS account and can't be changed after the load balancer is created.
 A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme [Info](#)
Scheme can't be changed after the load balancer is created.
 Internet-facing An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. [Learn more](#)
 Internal An internal load balancer routes requests from clients to targets using private IP addresses.

IP address type [Info](#)
Select the type of IP addresses that your subnets use.
 IPv4 Recommended for internal load balancers.
 Dualstack Includes IPv4 and IPv6 addresses.

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

The screenshot shows the 'Network mapping' section of the AWS Load Balancer configuration. It includes two tabs: 'VPC' and 'Mappings'. Under 'VPC', a VPC is selected. Under 'Mappings', four availability zones are listed, each with a checked checkbox. A red box highlights the list of four availability zones.

Availability Zone	Subnet
us-east-1a (use1-az2)	subnet-073ff5ef9a43683a0
us-east-1b (use1-az4)	subnet-02e0a980e794cbc1
us-east-1c (use1-az6)	subnet-0f330183ac197d551
us-east-1d (use1-az1)	subnet-0a9892baf955b4150

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

The screenshot shows the 'Security groups' section of the AWS Load Balancer configuration. It includes a 'Security groups' tab and a 'Listeners and routing' tab. In the 'Security groups' tab, a single security group is selected. In the 'Listeners and routing' tab, a listener for port 80 is configured to forward traffic to a target group.

Protocol	Port	Action
HTTP	80	Forward to: Select a target group

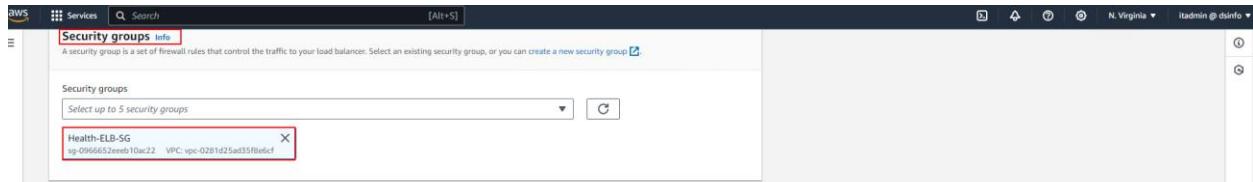
At this stage, specify a **name** for the Security Group.

The screenshot shows the 'Create security group' wizard in the AWS Management Console. The 'Basic details' section is highlighted with a red box. It contains fields for 'Security group name' (Health-ELB-SG) and 'Description' (Health-ELB-SG). Below this is a 'VPC Info' dropdown set to 'vpc-0281d25ad35f8e6cf'. The 'Inbound rules' section is shown below, indicating 'This security group has no inbound rules.' A 'Add rule' button is present.

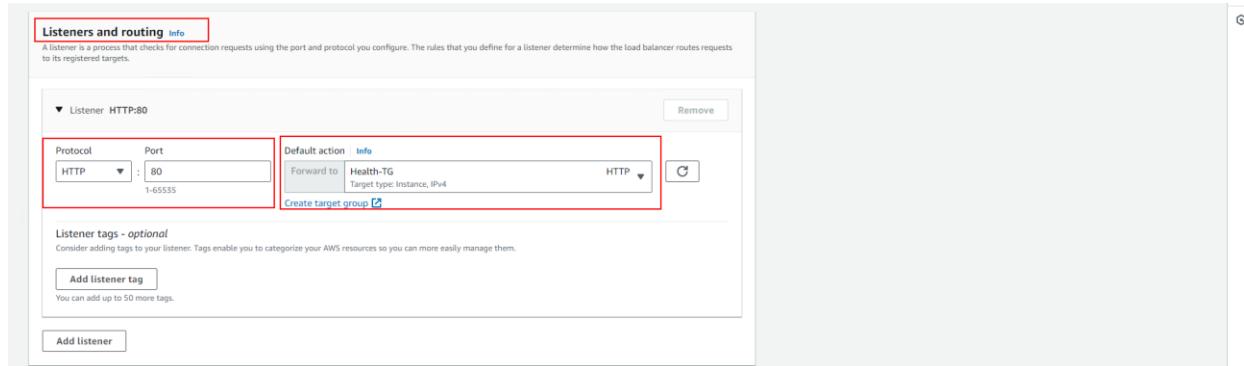
In the **Inbound Rule** section, you need to create the following rules, and then click on the **Create Security Group** button.

The screenshot shows the 'Create security group' wizard with the 'Inbound rules' section highlighted by a red box. Two rules are defined: one for port 80 (Custom TCP) with source 'Anywhere...' and another for port 80 (Custom TCP) with source '::/0'. Both rules have an optional description field and a 'Delete' button. A note at the bottom states: '⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.' Below this is a 'Tags - optional' section with a note about tags and a 'Create security group' button at the bottom right.

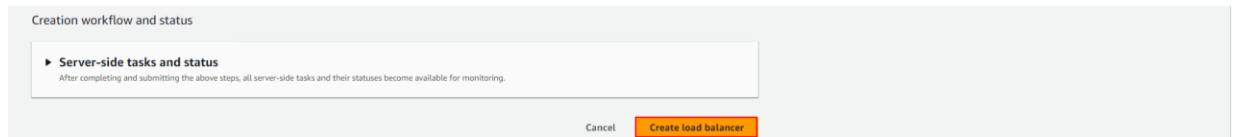
At this stage, select the **Security Group** you have created.



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.

The screenshot shows the AWS EC2 console with the 'Security Groups' page open. The left sidebar shows various services like Instances, Launch Templates, and Network & Security. Under Network & Security, 'Security Groups' is selected. The main area displays a table of security groups with columns for Name, Security group ID, Security group name, VPC ID, Description, and Owner. One row is selected, highlighted with a red box, and its details are shown in a modal below. The modal title is 'sg-090128606760a4c13 - launch-wizard-2'. It has tabs for 'Details', 'Inbound rules' (which is selected), 'Outbound rules', and 'Tags'. The 'Inbound rules' section shows a table with one rule: 'sgr-02767b08477d22a60'. The 'Edit inbound rules' button at the top right of this section is also highlighted with a red box.

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

The screenshot shows the 'Edit inbound rules' page for the 'sg-090128606760a4c13' security group. The 'Inbound rules' table has three rows: one for SSH (port 22), one for HTTP (port 80), and a new rule being added. The new rule's 'Type' is 'Custom TCP', 'Protocol' is 'TCP', 'Port range' is '80', and 'Source' dropdown shows 'sg-0966652eeeb10ac22'. A note at the bottom of the table says: '⚠️ Rules with source of 0.0.0.0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.' At the bottom right are 'Cancel', 'Preview changes', and 'Save rules' buttons. A red arrow points from the 'Health-ELB-SG' text to the 'Source' dropdown.

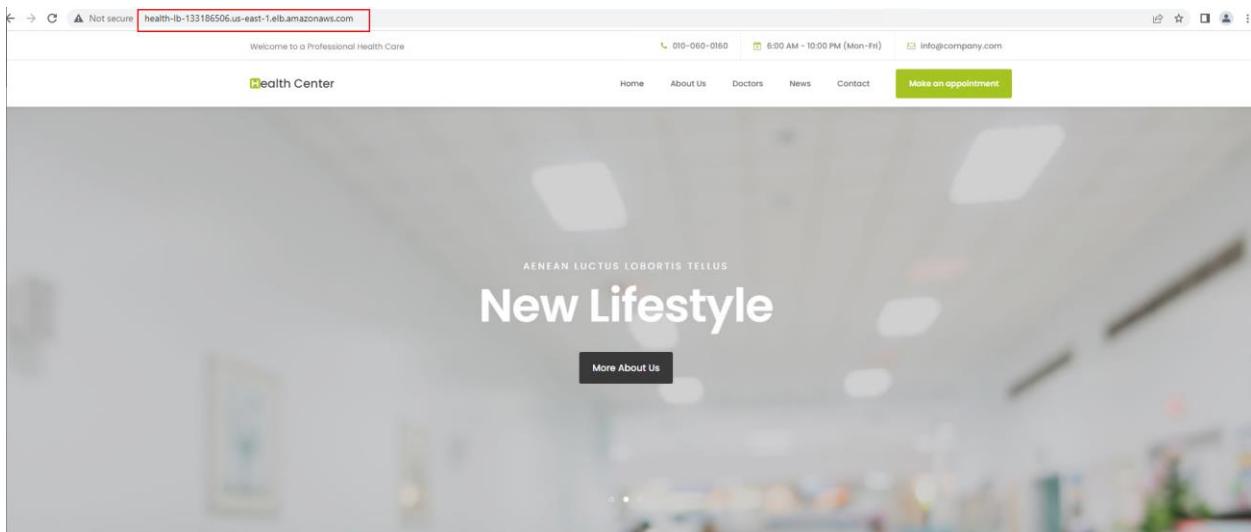
As shown in the image below, the status of the instances in the **Target Group** changes to **Healthy**.

The screenshot shows the AWS EC2 Target groups interface. On the left, a navigation menu includes options like Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces, Load Balancing, Load Balancers, Target Groups (selected), Trust Stores, Auto Scaling, and Auto Scaling Groups. The main area displays 'Target groups (1/1) info' for 'Health-TG'. The table shows one target: 'aws:elasticloadbalancing:target:1093418366137:loadbalancer/app/Health-LB/8dcf76d582a5612' with port 80, protocol HTTP, target type Instance, load balancer Health-LB, and VPC ID vpc-0281d25ad35f8e6cf. Below this, the 'Target group: Health-TG' details page shows 'Registered targets (2) info' with two entries: 'i-00be1e861e41bc292' (web002, port 80, zone us-east-1b, health status Healthy) and 'i-035da9911c7c52505' (web001, port 80, zone us-east-1b, health status Healthy). The status bar at the bottom indicates '© 2024, Amazon Web Services, Inc. or its affiliates.' and links for Privacy, Terms, and Cookie preferences.

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

The screenshot shows the AWS EC2 Load balancers interface. The left navigation menu is identical to the previous screenshot. The main area displays 'Load balancers (1/1)' for 'Health-LB'. The table shows one load balancer: 'Health-LB' with DNS name 'Health-LB-133186506.us-east-1.elb.amazonaws.com', state Active, VPC ID 'vpc-0281d25ad35f8e6cf', and 6 Availability Zones. Below this, the 'Load balancer: Health-LB' details page shows the load balancer ARN 'arn:aws:elasticloadbalancing:us-east-1:093418366137:loadbalancer/app/Health-LB/8dcf76d582a5612' and the DNS name 'Health-LB-133186506.us-east-1.elb.amazonaws.com (A Record)'. The status bar at the bottom indicates '© 2024, Amazon Web Services, Inc. or its affiliates.' and links for Privacy, Terms, and Cookie preferences.

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



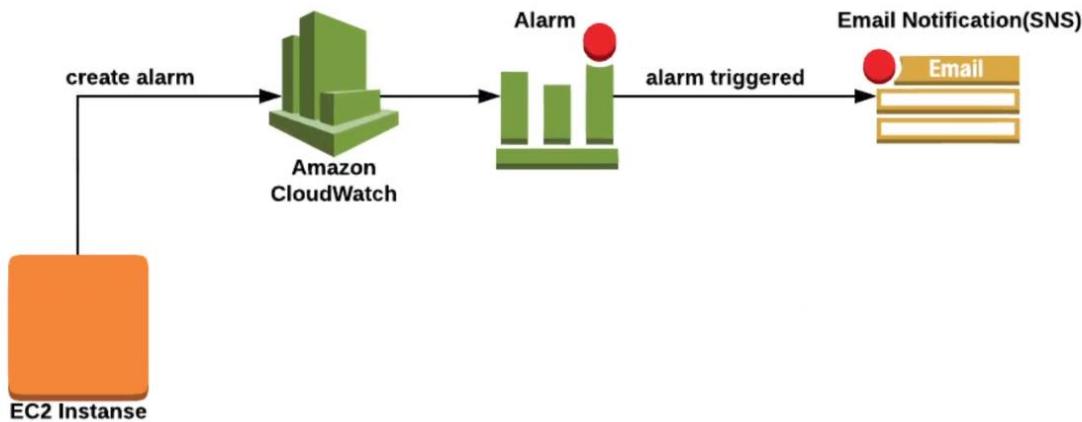
Introduction to AWS CloudWatch

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

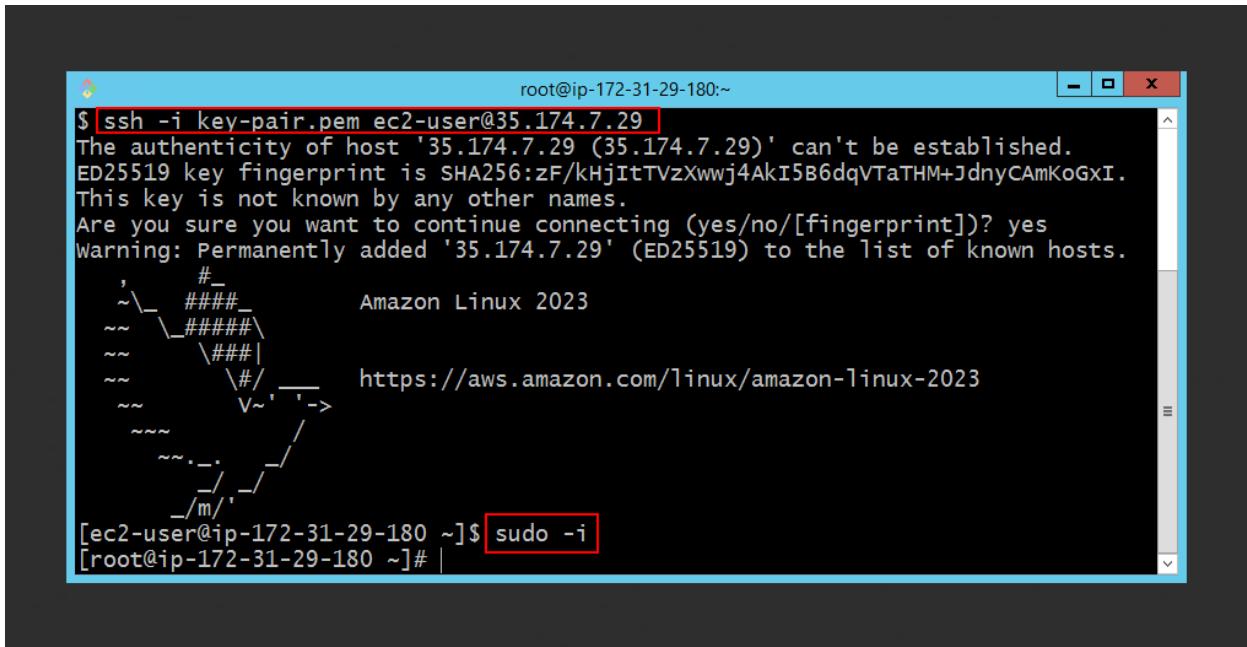
Introduction to AWS EFS (Elastic File System)

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**.

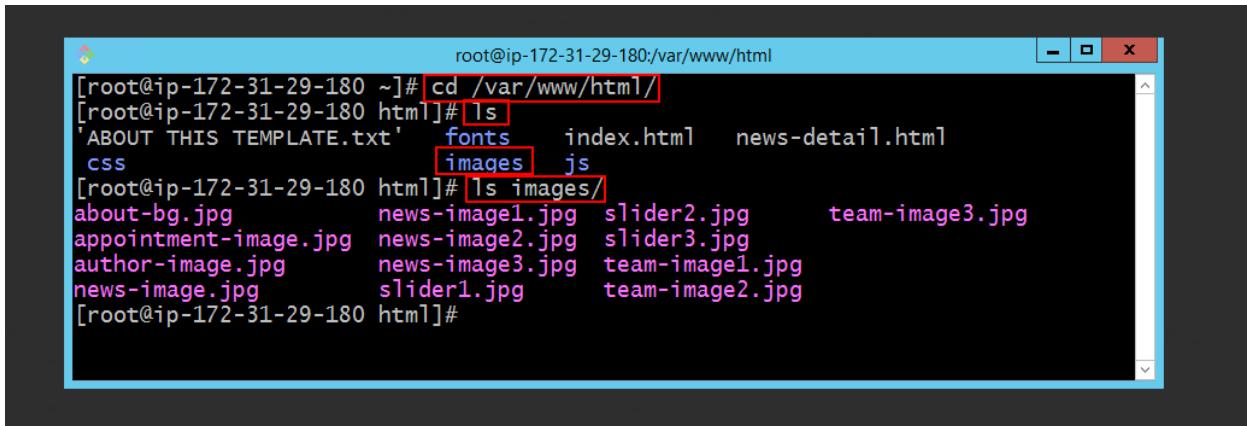


```
root@ip-172-31-29-180:~$ ssh -i key-pair.pem ec2-user@35.174.7.29
The authenticity of host '35.174.7.29 (35.174.7.29)' can't be established.
ED25519 key fingerprint is SHA256:zF/kHjItTVzXwwj4Aki5B6dqVTaTHM+JdnyCAmKoGxI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '35.174.7.29' (ED25519) to the list of known hosts.

          #_
         ~\_\ #####
         ~~ \_\#####\
         ~~   \|##|
         ~~     \|#/ __
         ~~       V~' _->
         ~~~      / \
         ~~~ .-. / \
         ~~~ /_ / \
         _/m/'

[ec2-user@ip-172-31-29-180 ~]$ sudo -i
[root@ip-172-31-29-180 ~]# |
```

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



```
root@ip-172-31-29-180:~# cd /var/www/html/
root@ip-172-31-29-180 html]# ls
ABOUT THIS TEMPLATE.txt  fonts  index.html  news-detail.html
css                      images  js
[root@ip-172-31-29-180 html]# ls images/
about-bg.jpg             news-image1.jpg  slider2.jpg    team-image3.jpg
appointment-image.jpg    news-image2.jpg  slider3.jpg
author-image.jpg         news-image3.jpg  team-image1.jpg
news-image.jpg           slider1.jpg    team-image2.jpg
[root@ip-172-31-29-180 html]#
```

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

The screenshot shows the AWS EC2 Security Groups page. The left sidebar includes sections for Instances, Images, Elastic Block Store, Network & Security, and Load Balancing. The main content area shows the details for the security group 'sg-0ed4d08504a257a3e - SG-ELB'. The 'Inbound rules' tab is selected, displaying three existing rules:

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
-	sgr-0f7fa9205dbd2ca85	IPv4	HTTP	TCP	80	84.32.10.4/32	-
-	sgr-0d8d2052de349965	-	HTTP	TCP	80	sg-0966652eeeb10ac...	-
-	sgr-0ecdf2d26b1d9d8cf	IPv4	SSH	TCP	22	0.0.0.0/0	-

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

The screenshot shows the 'Edit inbound rules' dialog box for the security group 'sg-0ed4d08504a257a3e - SG-ELB'. A new rule is being added with the following settings:

Security group rule ID	Type	Protocol	Port range	Source	Description
sgr-0f7fa9205dbd2ca85	HTTP	TCP	80	Custom (84.32.10.4/32)	
sgr-0d8d2052de349965	HTTP	TCP	80	Custom (sg-0966652eeeb10ac22)	
sgr-0ecdf2d26b1d9d8cf	SSH	TCP	22	Custom (0.0.0.0/0)	
-	NFS	TCP	2049	Custom (sg-0ed4d08504a257a3e)	SG-ELB

A warning message at the bottom states: "⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only." The 'Save rules' button is highlighted.

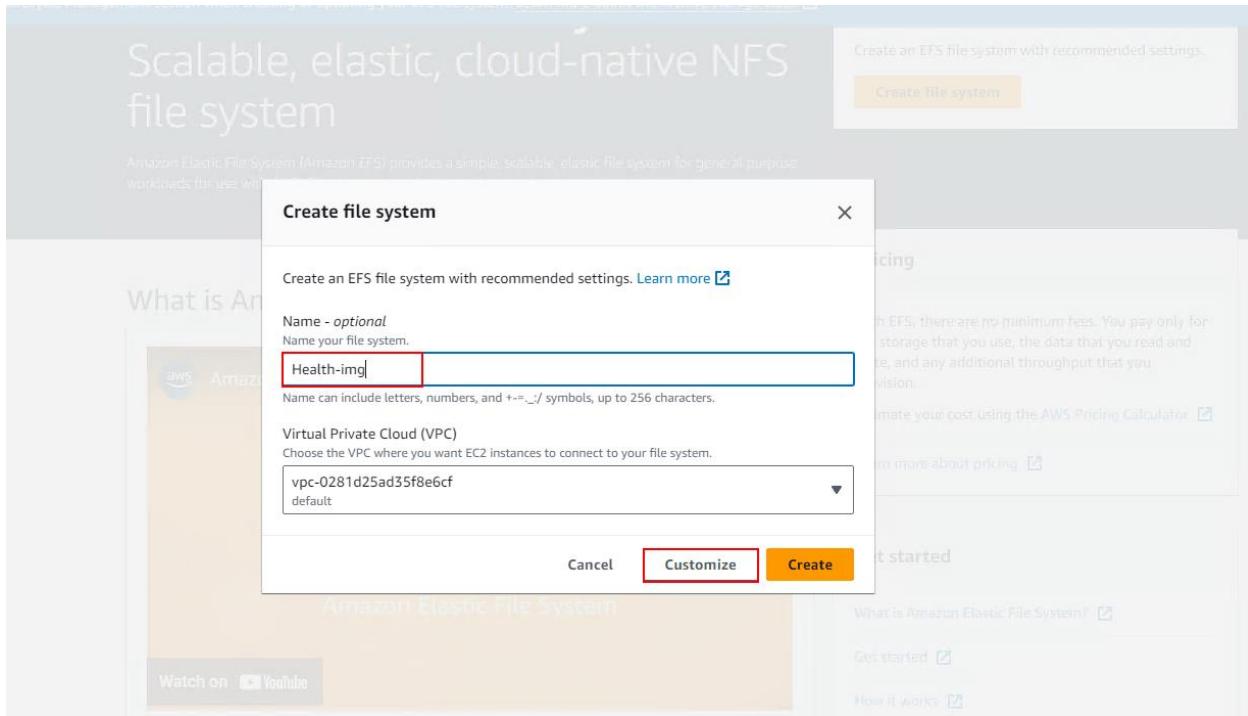
In the search bar, type **EFS**, then click on **EFS** from the results.

The screenshot shows the AWS Management Console search results for 'EFS'. In the 'Services' section, 'EFS' is listed under 'Features (19)' and is highlighted with a red box. Other features listed include DataSync, AWS Transfer Family, and MediaStore. To the right, the EC2 Instances page is displayed, showing two instances: 'us-east-1c' and 'us-east-1b' with their respective private and public IP addresses.

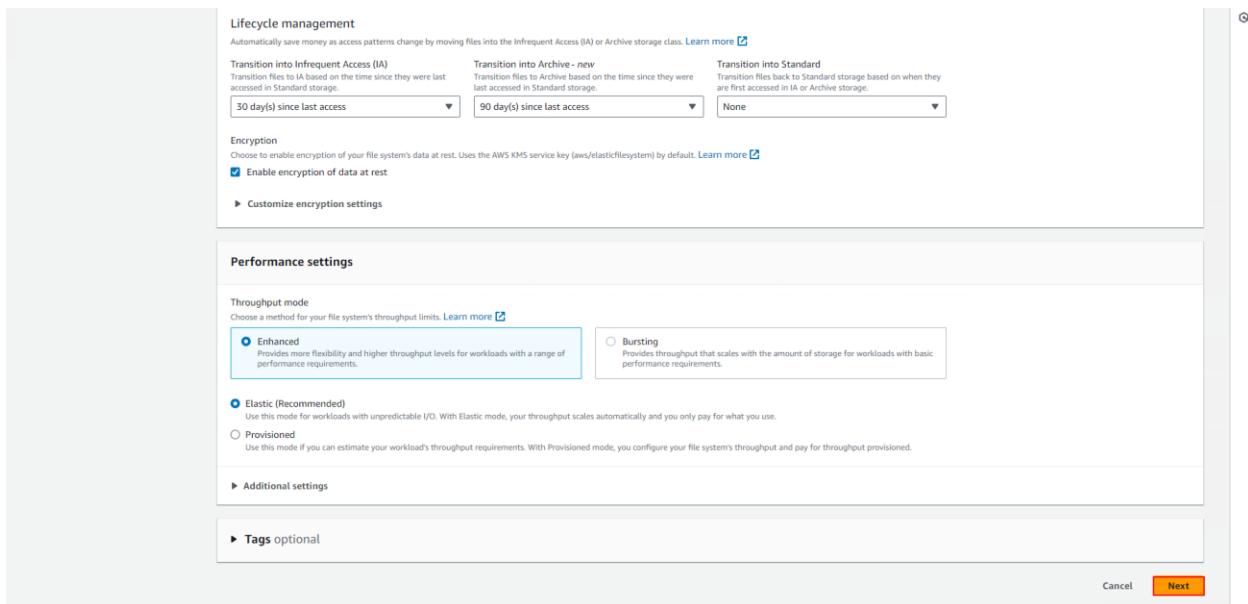
At this stage, click on the **Create File System** button.

The screenshot shows the Amazon Elastic File System (EFS) landing page. The main heading is 'Scalable, elastic, cloud-native NFS file system'. A prominent 'Create file system' button is located in the center. The left sidebar includes links for 'File systems', 'Access points', 'AWS Backup', 'AWS DataSync', and 'AWS Transfer'. The right side features sections for 'Pricing', 'Get started', and 'More resources'.

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**.

Step 1
File system settings

Step 2
Network access

Step 3 - optional
File system policy

Step 4
Review and create

Network access

Virtual Private Cloud (VPC) [Learn more](#) Choose the VPC where you want EC2 instances to connect to your file system.

vpc-0281d25ad35f8e6cf default

Mount targets

A mount target provides an NFSv4 endpoint at which you can mount an Amazon EFS file system. We recommend creating one mount target per Availability Zone. [Learn more](#)

Availability zone	Subnet ID	IP address
us-east-1a	subnet-073ff5ef9a43683a0	Automatic
us-east-1b	subnet-02e0a980e794cbc1f	Automatic
us-east-1c	subnet-0f330183ac197d551	Automatic
us-east-1d	subnet-0a9892baf955b4150	Automatic
us-east-1f	subnet-0be6450dc0522748d	Automatic

Security groups

Choose security groups [Remove](#)

sg-0ed4d08504a257a3e SG-ELB

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

Introducing Amazon EFS Archive

Amazon Elastic File System (EFS) now offers a new Archive storage class, which is cost-optimized for long-lived data that is accessed only a few times per year or less. To use Archive, choose one of the available "Transition into Archive" options in the Lifecycle Management section when creating or updating your EFS file system. [Learn more about the Archive storage class.](#)

Amazon EFS > File systems > Create

Step 1
File system settings

Step 2
Network access

Step 3 - optional
File system policy

Step 4
Review and create

File system policy - optional

Policy options

Select one or more of these common policy options, or create a custom policy using the editor. [Learn more](#)

Prevent root access by default*

Enforce read-only access by default*

Prevent anonymous access

Enforce in-transit encryption for all clients

* Identity-based policies can override these default permissions.

Grant additional permissions

Policy editor (JSON)

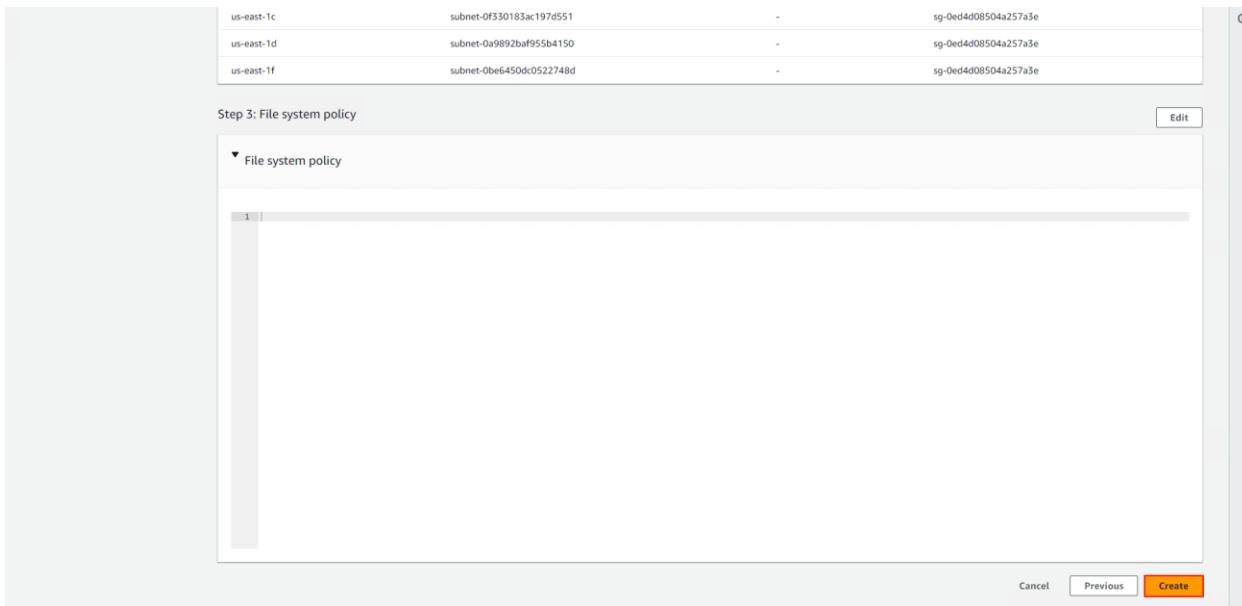
1 |

Clear

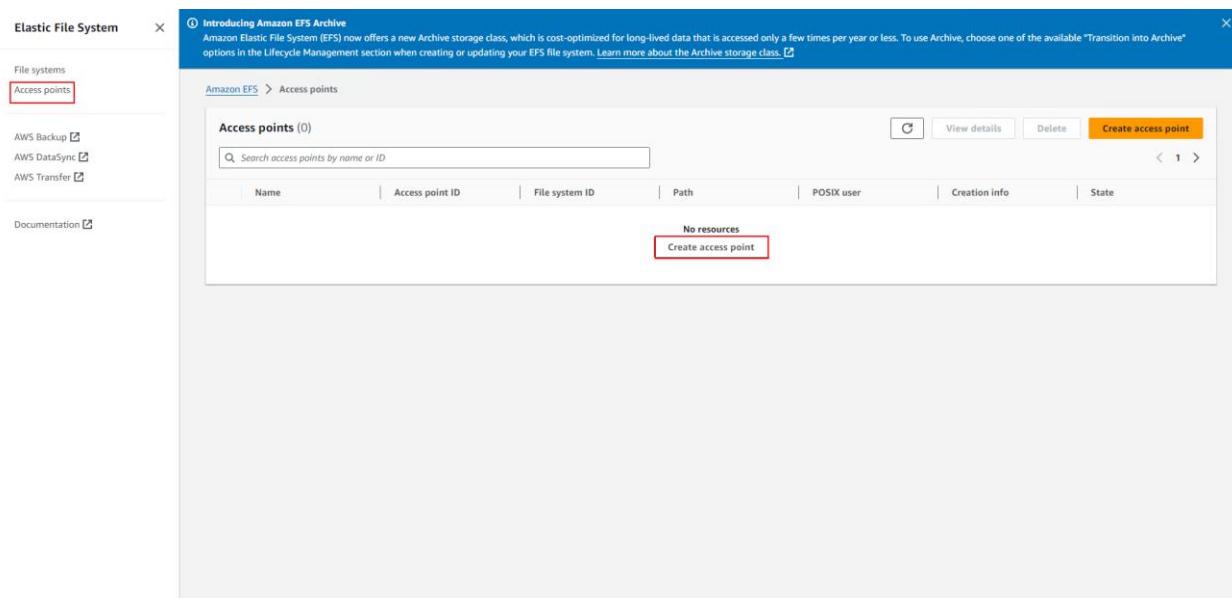
Manual changes will prevent the use of the policy options on the left until the editor is cleared.

Cancel Previous Next

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.

[X](#)

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](#)

[X](#)

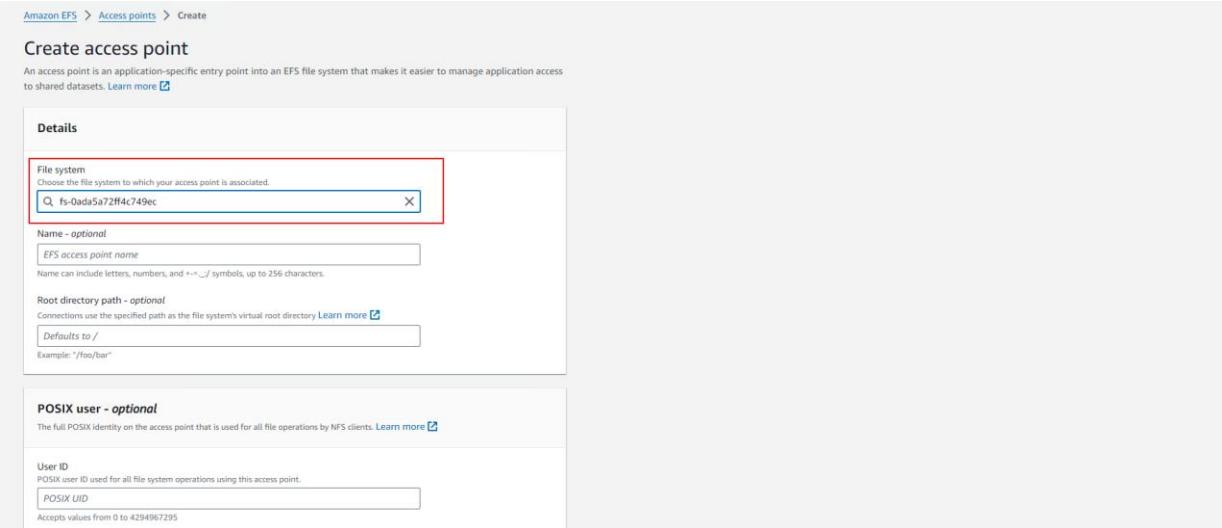
Example: `/foo/bar`

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.

[X](#)

Accepts values from 0 to 4294967295



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

The screenshot shows the 'Root directory creation permissions - optional' section of the AWS EFS Create Access Point wizard. It includes fields for Owner user ID (POSIX user ID to apply to path), Owner group ID (POSIX group ID to apply to path), and Access point permissions (example: "0755"). Below this is the 'Tags - optional' section, which allows adding key-value pairs. A red box highlights the 'Create access point' button at the bottom right.

Root directory creation permissions - optional

EFS will automatically create the specified root directory with these permissions if the directory does not already exist. [Learn more](#)

Owner user ID
Owner user ID for the access point's root directory, if the directory does not already exist.

Accepts values from 0 to 4294967295

Owner group ID
Owner group ID for the access point's root directory, if the directory does not already exist.

Accepts values from 0 to 4294967295

Access point permissions
POSIX permissions to apply to the root directory path

An octal number representing the file's mode bits.

Tags - optional

Add tags to associate key-value pairs to your resource. [Learn more](#)

Tag key Tag value - optional [Remove tag](#)

[Add tag](#)

You can add 49 more tag(s)

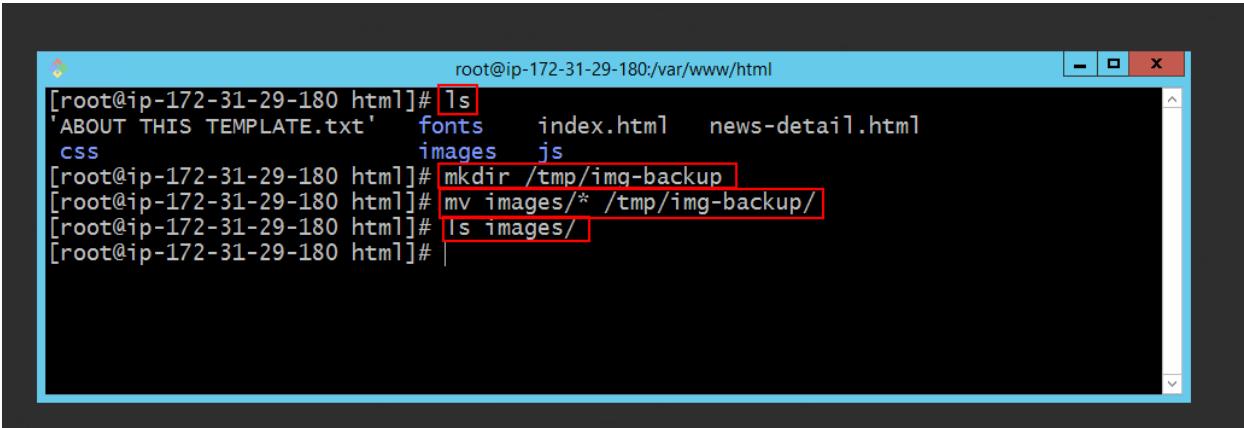
[Cancel](#) [Create access point](#)

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

The screenshot shows a terminal window on an Amazon Linux instance. The user is running the command `sudo yum install -y amazon-efs-utils`. The output shows the package being installed along with its dependencies, `stunnel`.

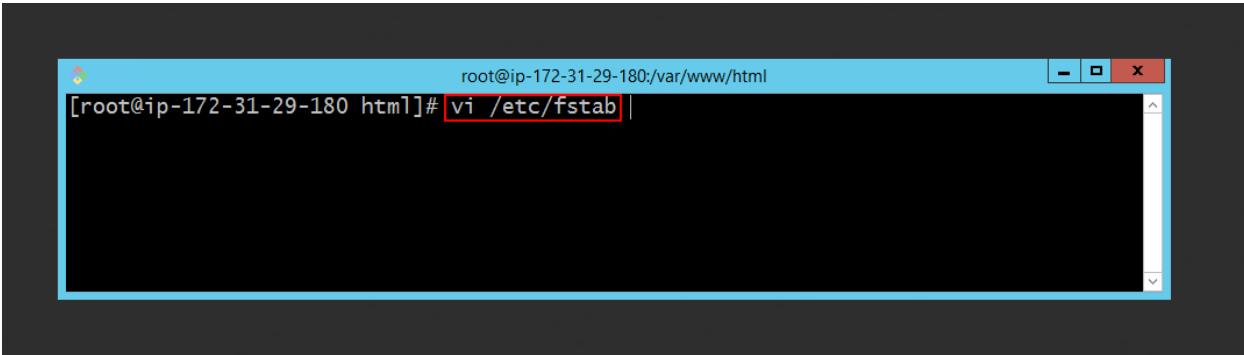
```
[root@ip-172-31-29-180 html]# sudo yum install -y amazon-efs-utils
Last metadata expiration check: 1:06:30 ago on Sat Mar 23 08:04:13 2024.
Dependencies resolved.
=====
 Package           Arch      Version       Repository     Size
=====
 Installing:
  amazon-efs-utils    noarch   1.35.2-1.amzn2023  amazonlinux   55 k
 Installing dependencies:
  stunnel            x86_64   5.58-1.amzn2023.0.2  amazonlinux  156 k
Transaction Summary
```

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



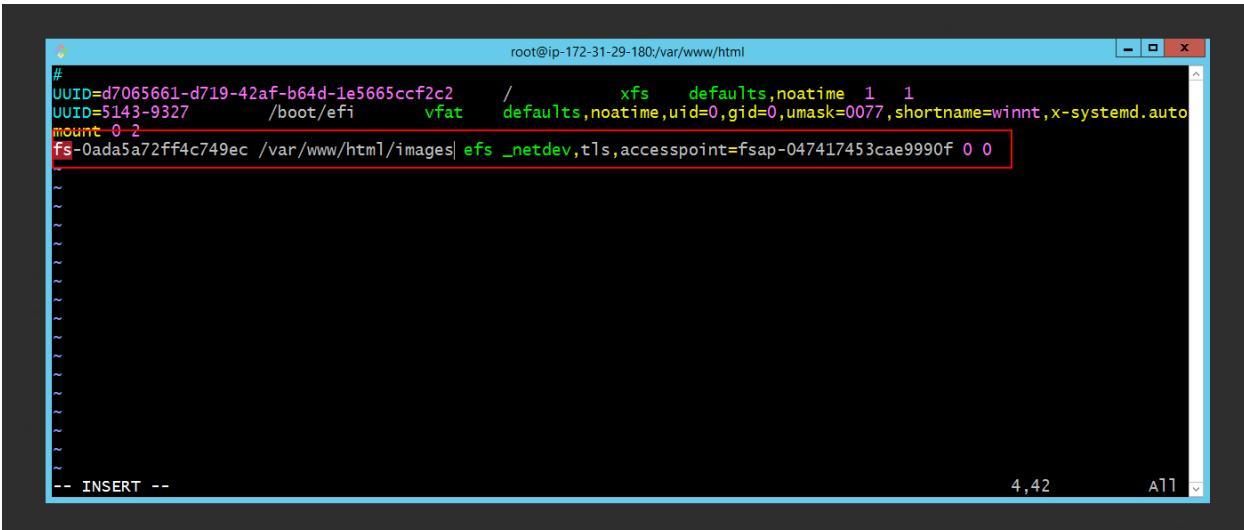
```
root@ip-172-31-29-180 html]# ls
'ABOUT THIS TEMPLATE.txt'  fonts  index.html  news-detail.html
css  images  js
[root@ip-172-31-29-180 html]# mkdir /tmp/img-backup
[root@ip-172-31-29-180 html]# mv images/* /tmp/img-backup/
[root@ip-172-31-29-180 html]# ls images/
[root@ip-172-31-29-180 html]#
```

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



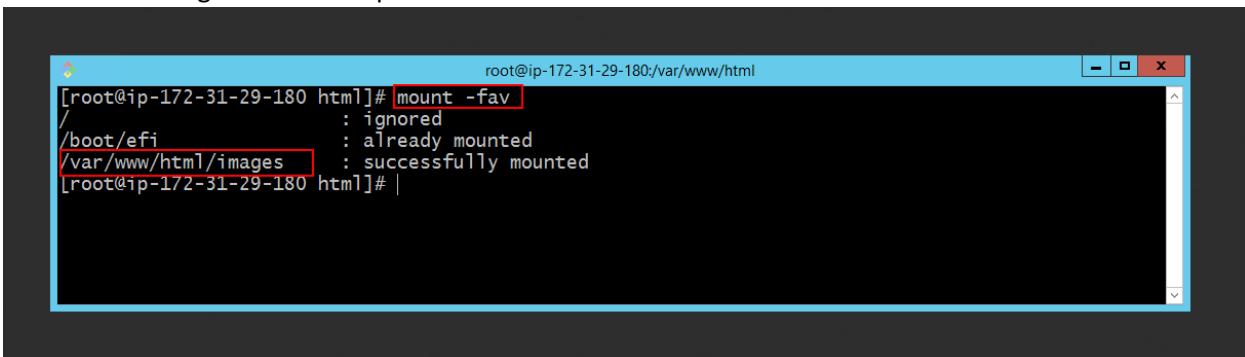
```
root@ip-172-31-29-180 html]# vi /etc/fstab |
```

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



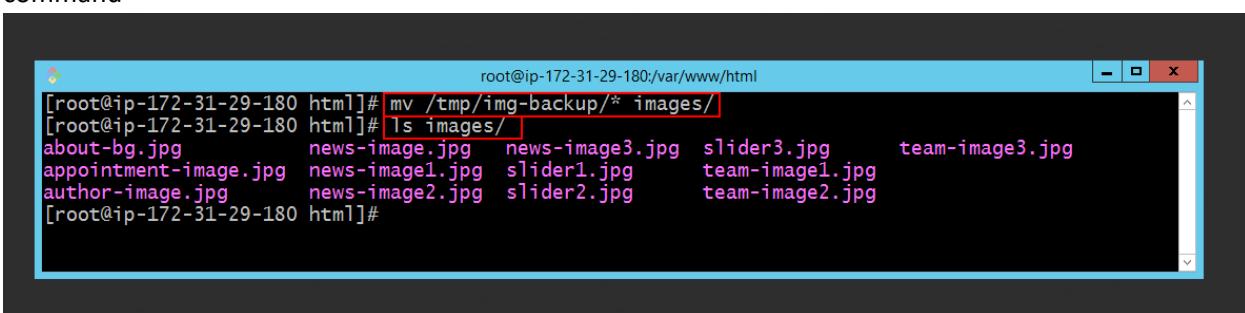
```
#  
UUID=d7065661-d719-42af-b64d-1e5665ccf2c2      /          xfs  defaults,noatime 1  1  
UUID=5143-9327        /boot/efi      vfat  defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.mount=-0_2  
fs-0ada5a72ff4c749ec /var/www/html/images| efs _netdev,tls,accesspoint=fsap-047417453cae9990f 0 0
```

Use the following command to perform the mount



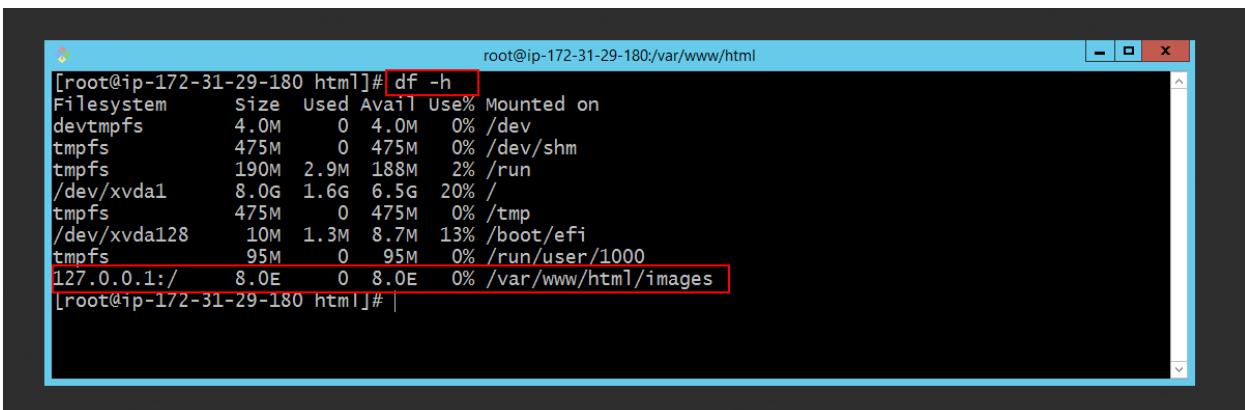
```
root@ip-172-31-29-180 html]# mount -f
/                         : ignored
/boot/efi                  : already mounted
/var/www/html/images        : successfully mounted
[root@ip-172-31-29-180 html]# |
```

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



```
root@ip-172-31-29-180 html]# mv /tmp/img-backup/* images/
[root@ip-172-31-29-180 html]# ls images/
about-bg.jpg      news-image.jpg   news-image3.jpg  slider3.jpg    team-image3.jpg
appointment-image.jpg news-image1.jpg slider1.jpg   team-image1.jpg
author-image.jpg   news-image2.jpg slider2.jpg   team-image2.jpg
[root@ip-172-31-29-180 html]#
```

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



```
root@ip-172-31-29-180 html]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        4.0M   0  4.0M  0% /dev
tmpfs          475M   0  475M  0% /dev/shm
tmpfs          190M  2.9M 188M  2% /run
/dev/xvda1     8.0G  1.6G  6.5G 20% /
tmpfs          475M   0  475M  0% /tmp
/dev/xvda128   10M  1.3M  8.7M 13% /boot/efi
tmpfs          95M   0  95M  0% /run/user/1000
127.0.0.1:/    8.0E   0  8.0E  0% /var/www/html/images
[root@ip-172-31-29-180 html]# |
```

Introduction to AWS Auto Scaling

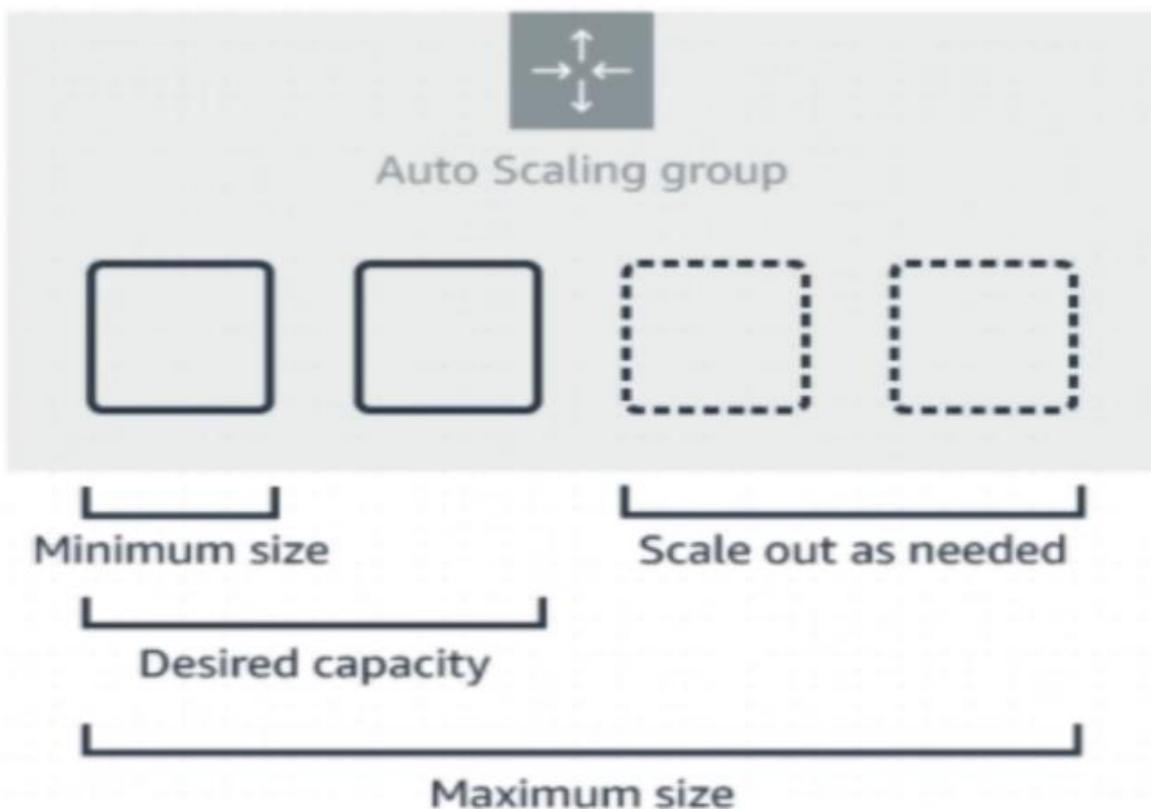
Auto Scaling is a service that can automatically **monitor** and **adjust compute resources** to improve the **performance** of applications hosted on your **AWS** environment.

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.

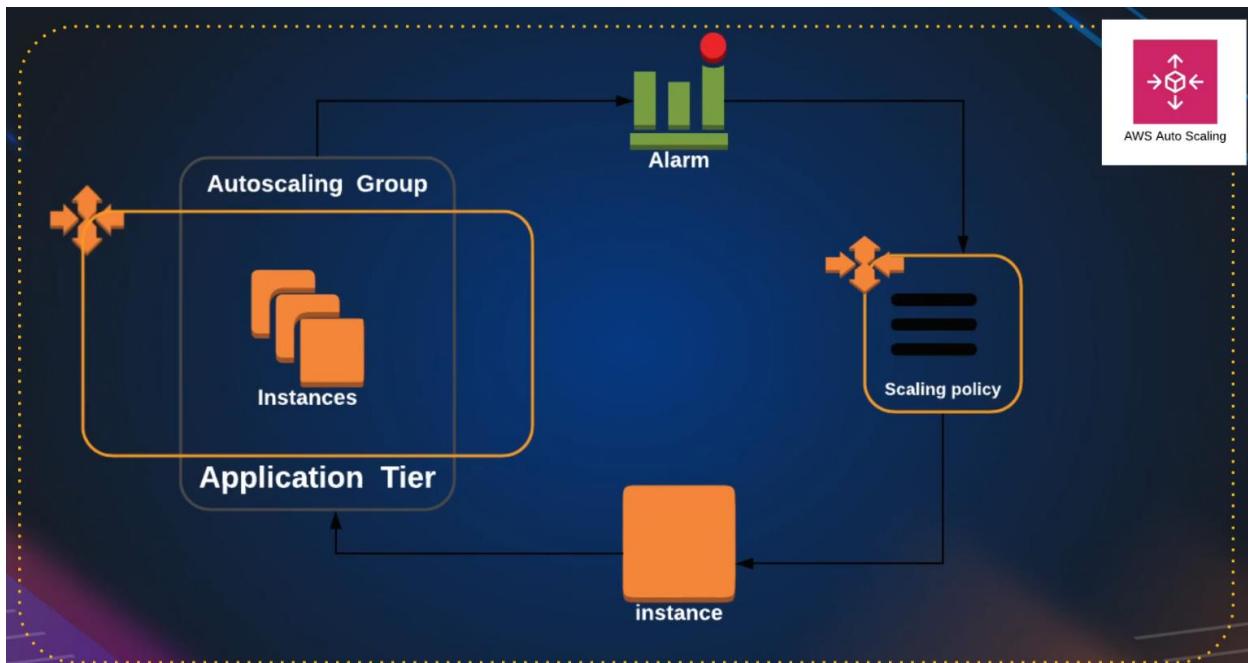


When you create **Auto Scaling**, you can define the **Minimum Size** of the Auto Scaling Group. For example, you can specify **1 instance** as the **minimum instance** in the Auto Scaling Group.

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.

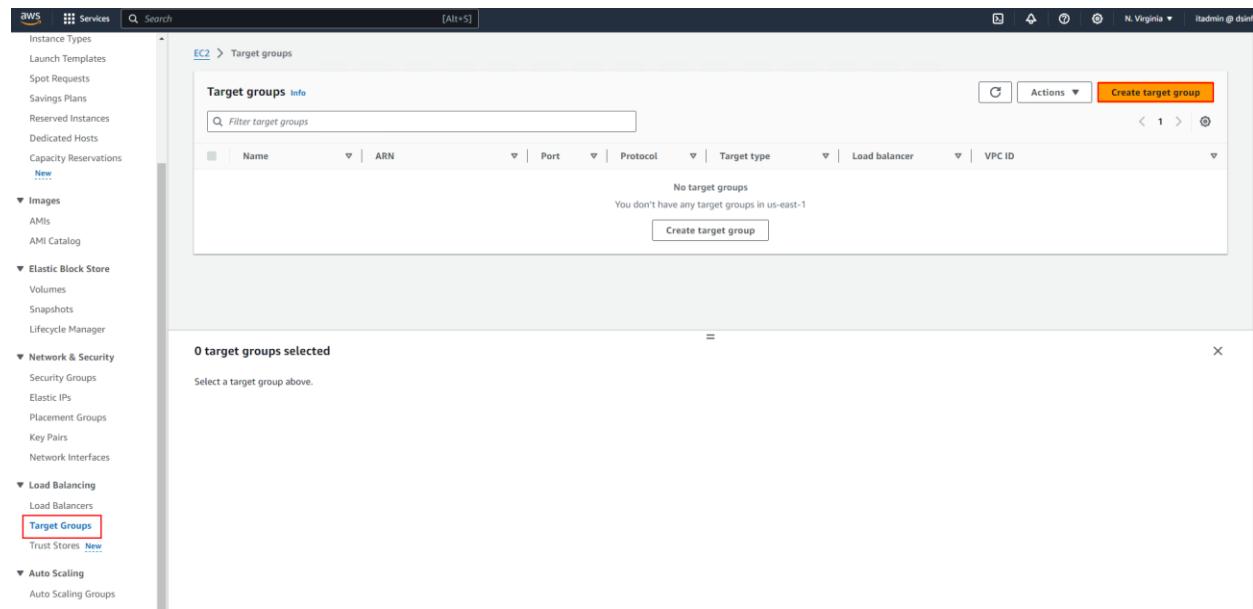
Therefore, when the **Minimum Instance** is set to **1**, during termination, at least **one instance** will remain in the group, and the rest will be terminated. As a result, you'll always have the number of **Desired Instances** running in your **Auto Scaling Group**.

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



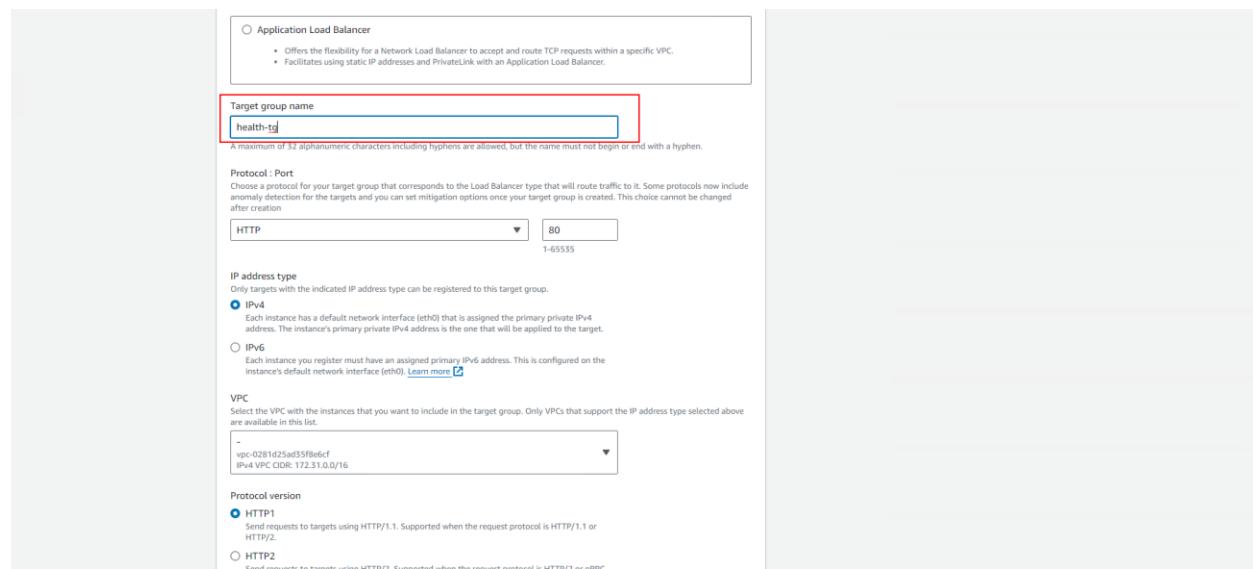
How to Create an AWS Auto Scaling Group

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



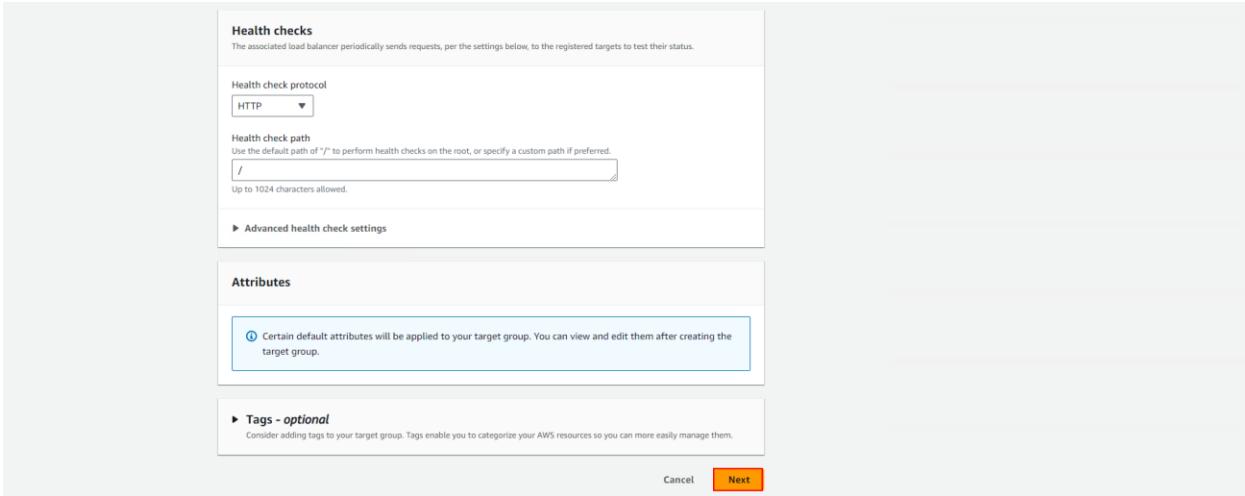
The screenshot shows the AWS EC2 console with the 'Target Groups' section selected under 'Load Balancing'. The main pane displays a table with columns for Name, ARN, Port, Protocol, Target type, Load balancer, and VPC ID. A message at the top right states 'No target groups' and 'You don't have any target groups in us-east-1'. At the bottom right of the table area is a 'Create target group' button. The left sidebar lists various AWS services like Instance Types, Launch Templates, and Auto Scaling.

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

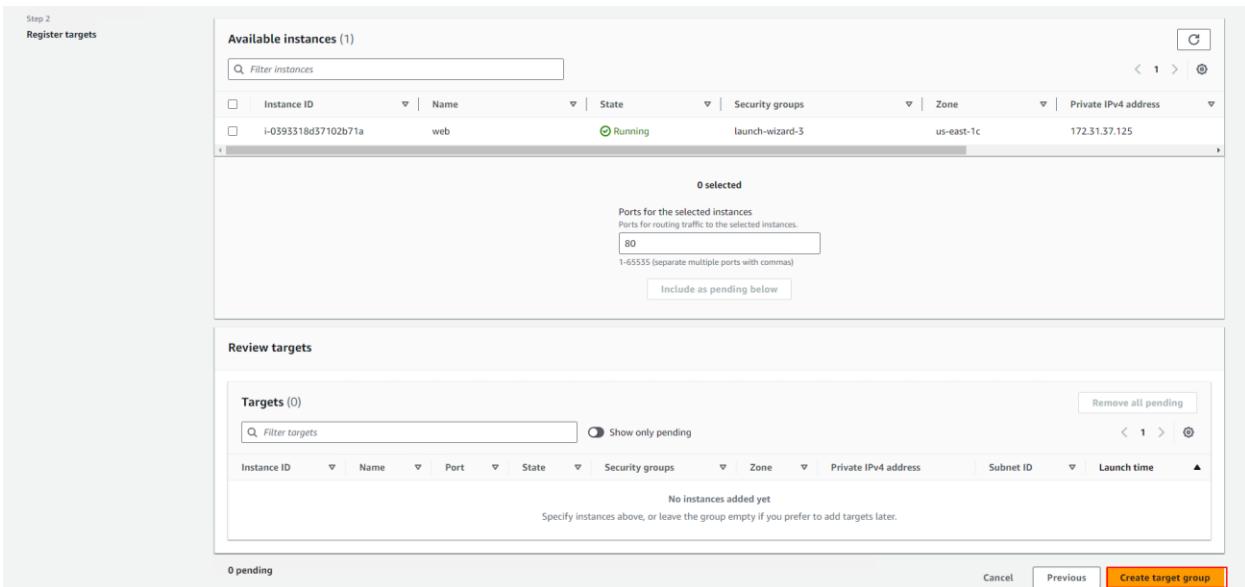


The screenshot shows the 'Create Target Group' configuration page for an Application Load Balancer. The 'Target group name' field is highlighted with a red box and contains the value 'health-tg'. Other fields shown include 'Protocol: Port' (HTTP, port 80), 'IP address type' (IPv4 selected), 'VPC' (a single VPC entry selected), and 'Protocol version' (HTTP1 selected). The page also includes descriptive text and links for network load balancing and target group creation.

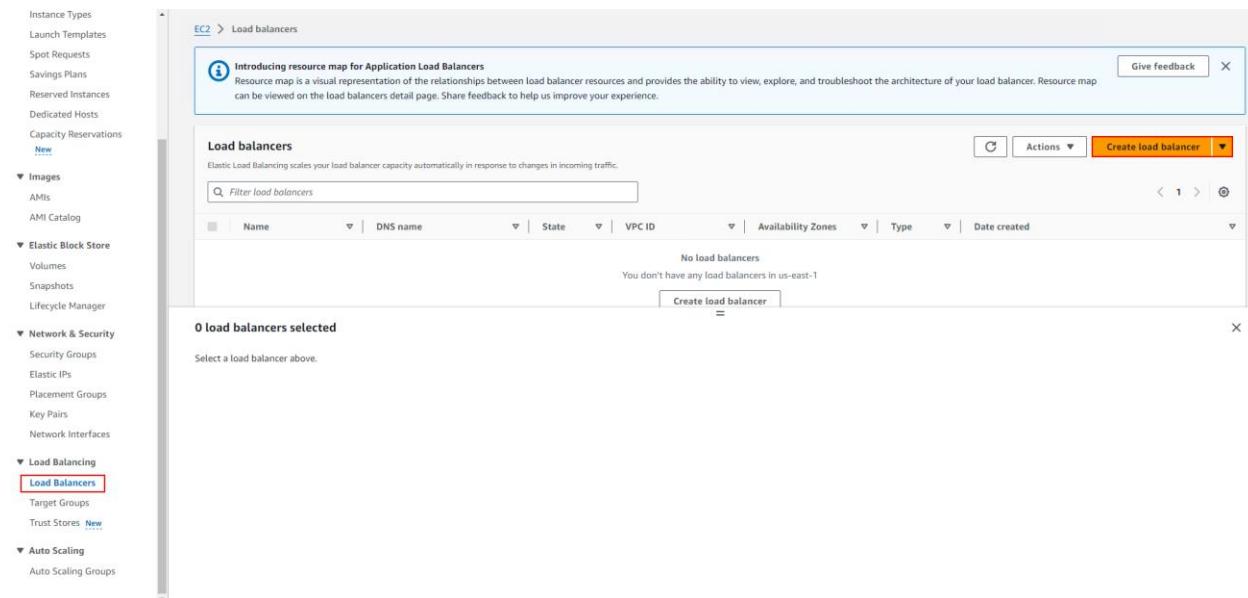
At this stage, click on the **Next** button.



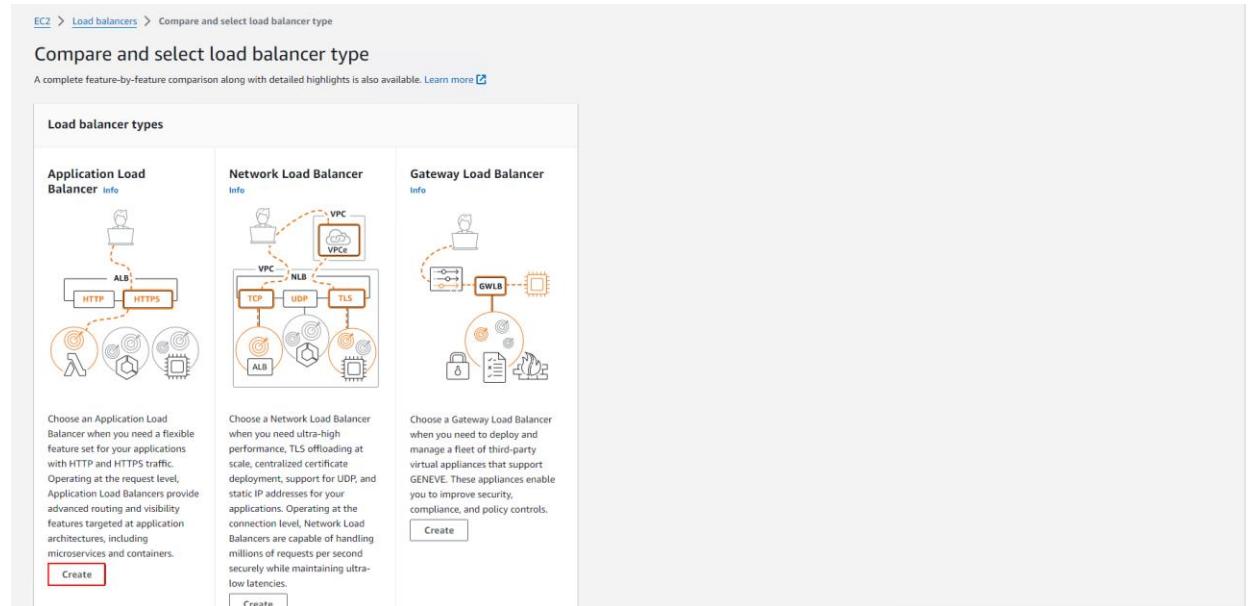
Then, at this stage, click on the **Create Target Group** button.



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.

The screenshot shows the 'Create Application Load Balancer' wizard. In the 'Basic configuration' step, the 'Load balancer name' field is filled with 'Health-ELB'. Below it, the 'Scheme' section shows 'Internet-facing' selected. Under 'IP address type', 'IPv4' is chosen. The 'Network mapping' section indicates traffic will be routed to selected subnets.

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

The screenshot shows the 'Network mapping' section where six Availability Zones (us-east-1a through us-east-1f) are selected. Each row includes a checkbox, the zone name, a 'Subnet' dropdown, and an 'IPv4 address' field indicating it's assigned by AWS.

Availability Zone	Subnet	IPv4 address
us-east-1a (use1-az2)	subnet-073ff5ef9a43683a0	Assigned by AWS
us-east-1b (use1-az4)	subnet-02e0a980e794cbc1	Assigned by AWS
us-east-1c (use1-az6)	subnet-0f30183ac197d551	Assigned by AWS
us-east-1d (use1-az1)	subnet-0a9892ba955b4150	Assigned by AWS
us-east-1e (use1-az3)	subnet-05af0b7eee6bb7b6	Assigned by AWS
us-east-1f (use1-az5)	subnet-0be6450dc0522748d	

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.

The screenshot shows the AWS Load Balancer configuration interface. In the 'Security groups' section, 'SG-ELB' is selected. In the 'Listeners and routing' section, a new listener 'HTTP:80' is being configured. The 'Protocol' is set to 'HTTP' and the 'Port' is '80'. The 'Default action' dropdown is set to 'Forward to' 'health-tg', which is highlighted with a red box. Below this, there is a 'Create target group' button. The 'Listeners tags - optional' section contains a 'Add listener tag' button. At the bottom left is a 'Add listener' button.

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

The screenshot shows the 'Review and confirm your configurations' step. It displays the following details:

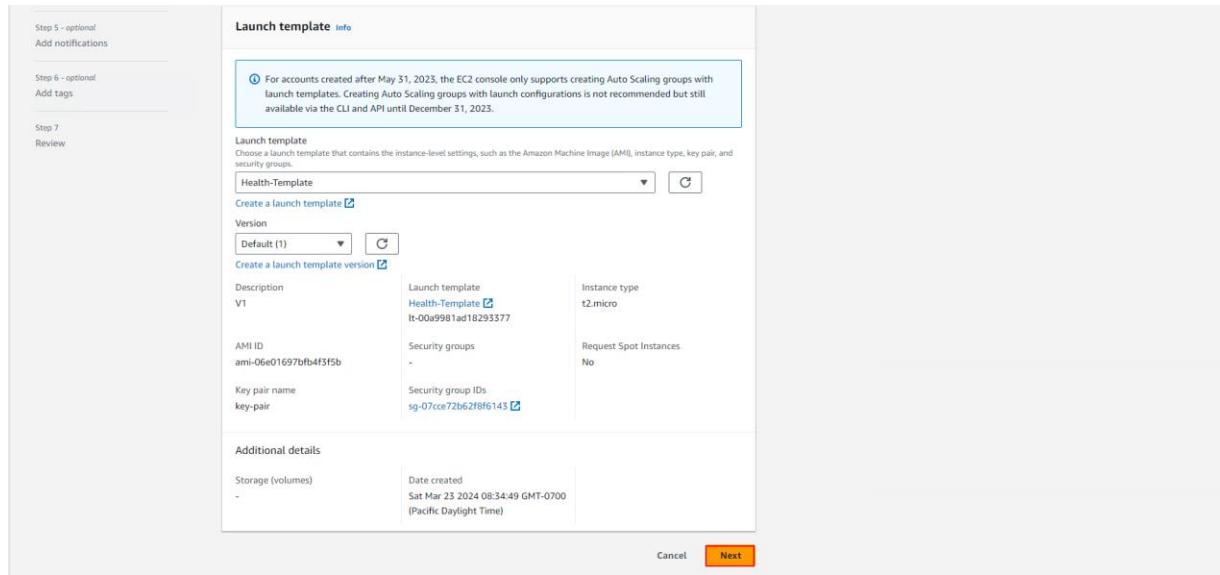
- Basic configuration:** Health-ELB, Internet-facing, IPv4.
- Security groups:** SG-ELB (selected), VPC: ypc-0281d25ad35f8e6cf.
- Network mapping:** VPC: ypc-0281d25ad35f8e6cf, Subnets: us-east-1a, us-east-1b, us-east-1c, us-east-1d, us-east-1e, us-east-1f.
- Listeners and routing:** Listener 'HTTP:80' is mapped to 'health-tg'.
- Service integrations:** AWS WAF: None, AWS Global Accelerator: None.
- Tags:** None.
- Attributes:** A note states: 'Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer.'
- Creation workflow and status:** A note states: 'After completing and submitting the above steps, all server-side tasks and their statuses become available for monitoring.'

At the bottom right, there are 'Cancel' and 'Create load balancer' buttons, with 'Create load balancer' being highlighted.

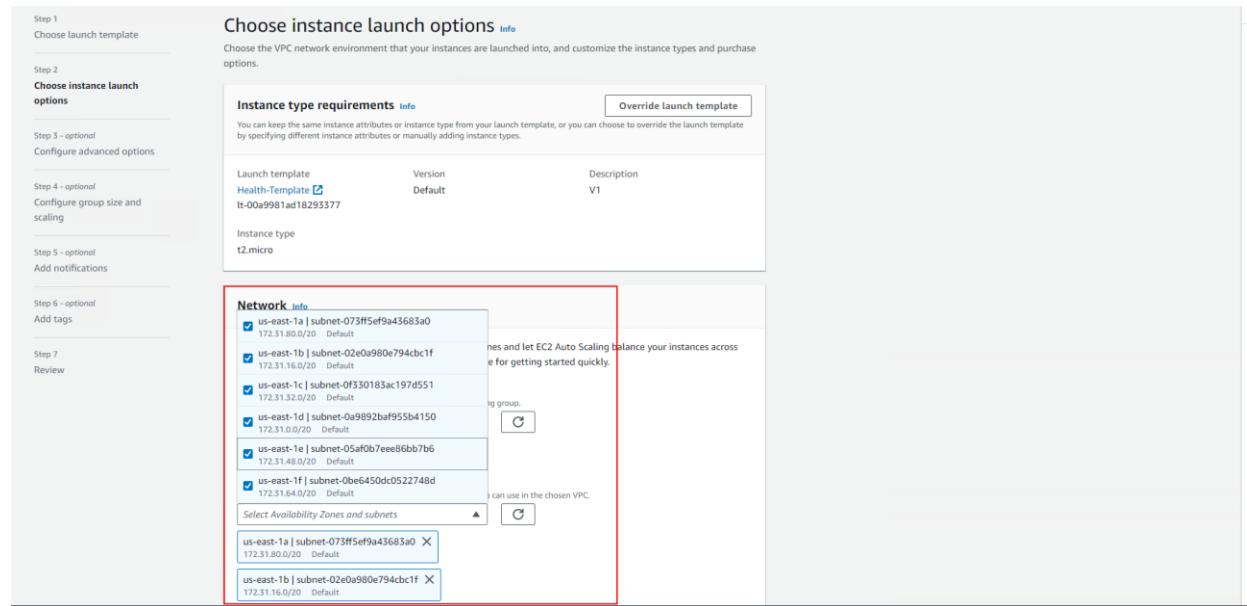
At this stage, click on the **Auto Scaling Groups** section, and then click on the **Create Auto Scaling Group** button.

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

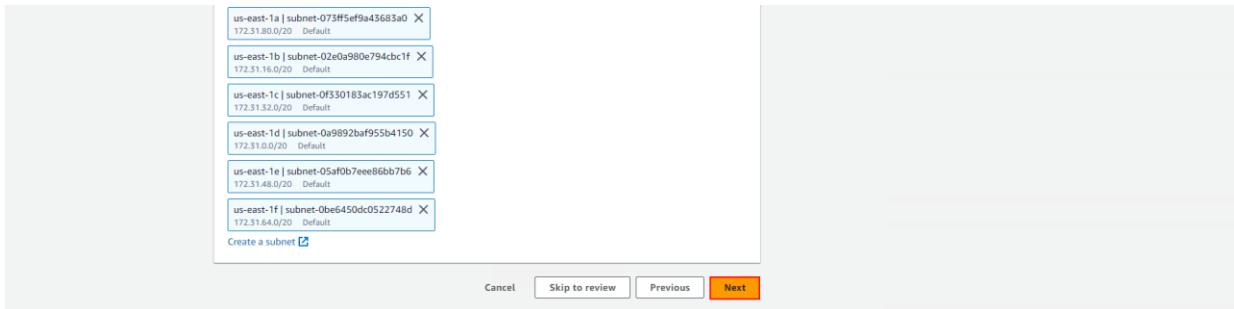
At this stage, click on the **Next** button.



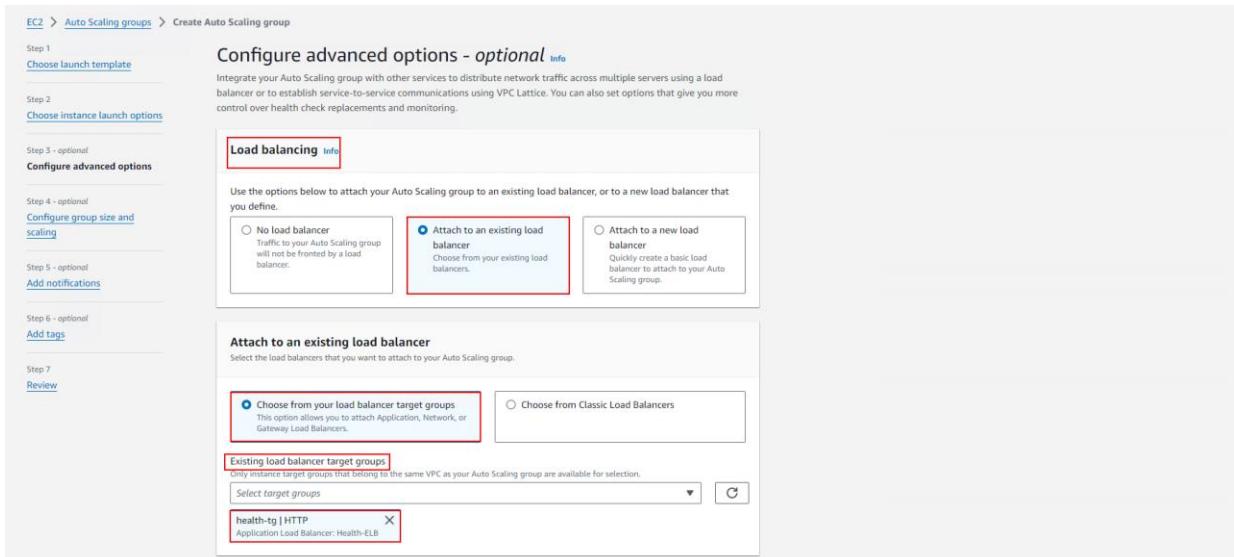
At this stage, select **all Availability Zones** to enhance availability.



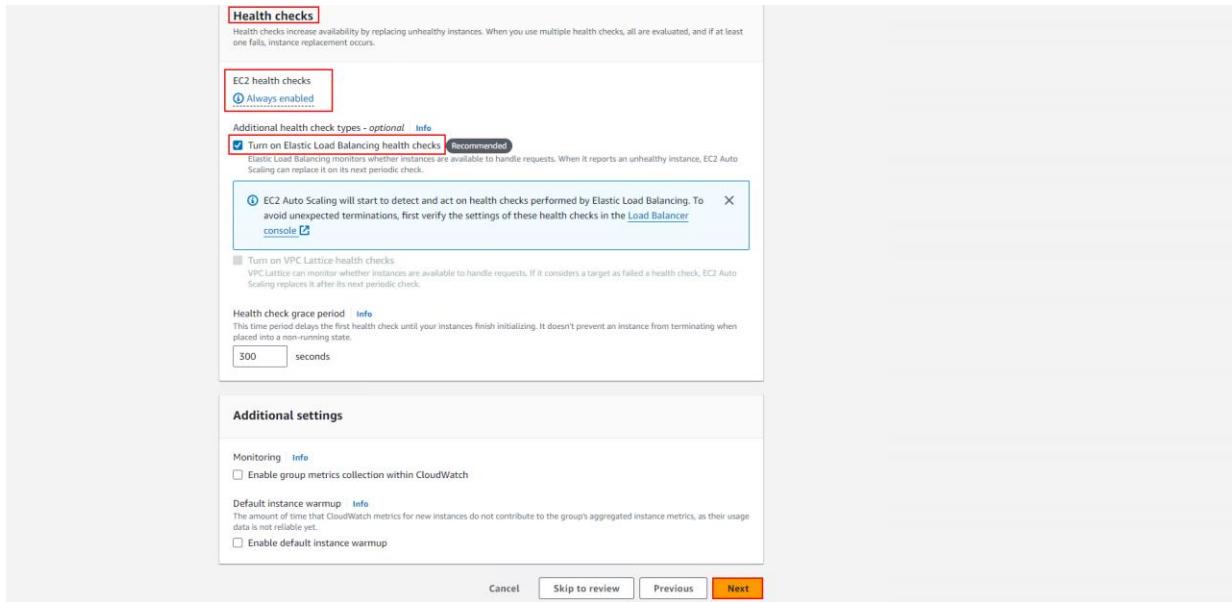
Then, at this stage, click on the **Next** button.



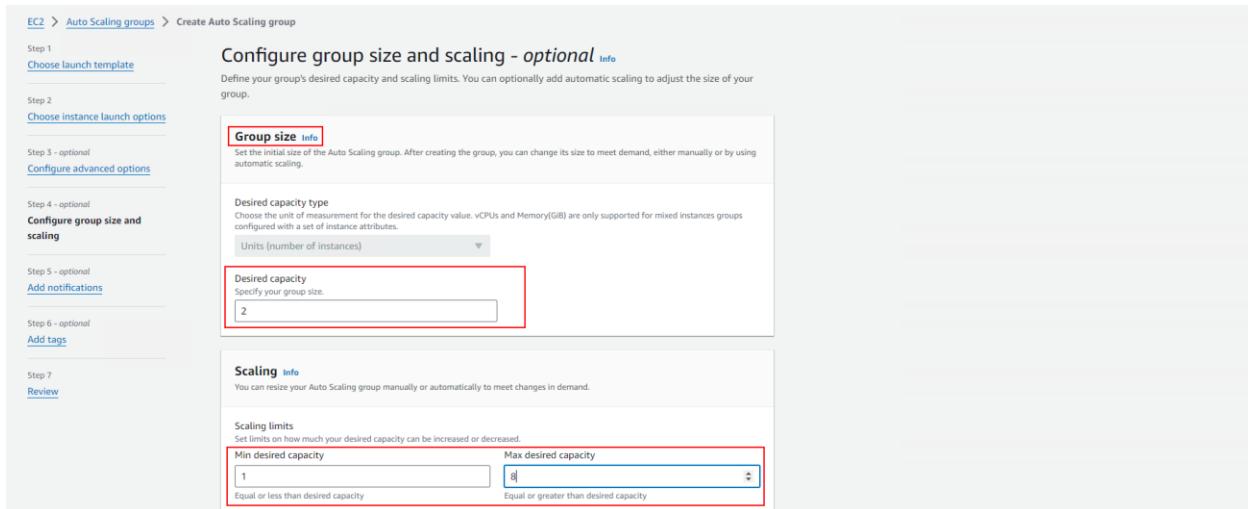
At this stage, in the **Load Balancing** section, select the option **Attach to an existing load balancer**, and then specify the **Target Group**.



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



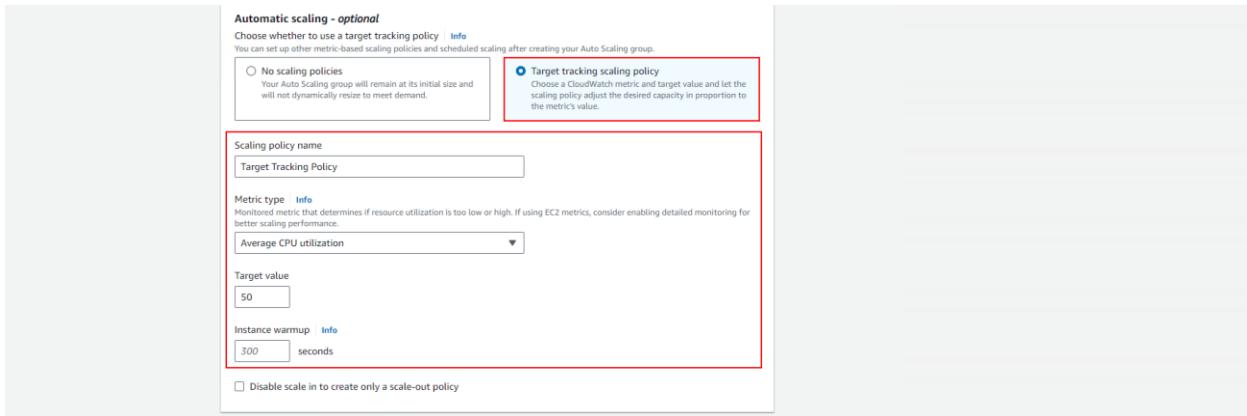
At this stage, you need to configure the **Group Size** settings, as shown in the image below.



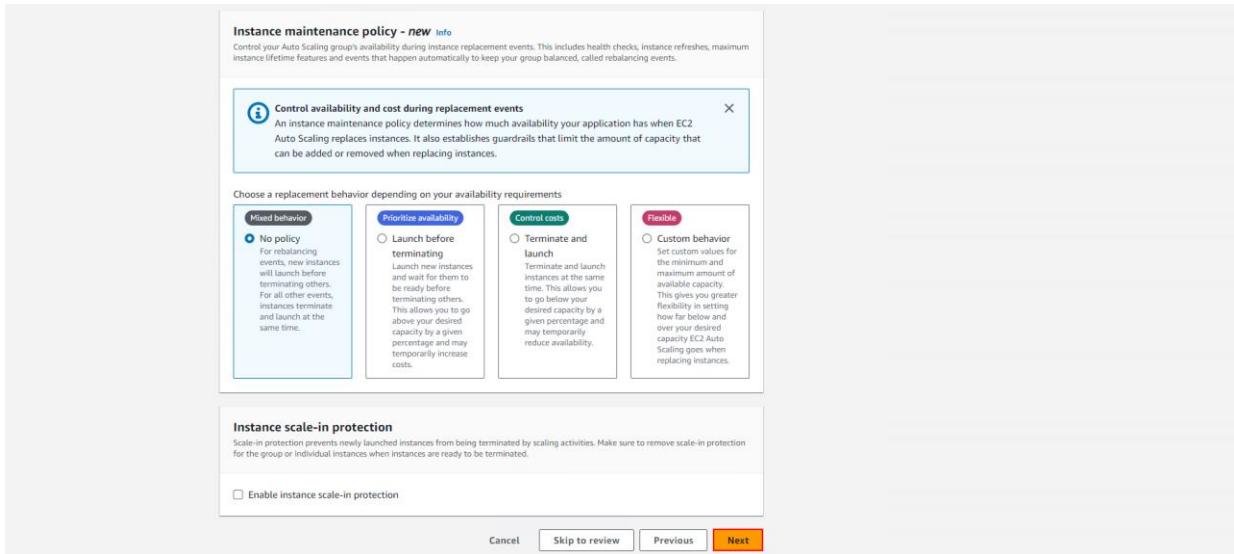
At this stage, select the **Target Tracking Scaling Policy** option, then set the **Metric Type** to **Average CPU Utilization**, and set the **Target Value** to **50%**.

This means that if the average CPU utilization exceeds 50%, Auto Scaling will increase the

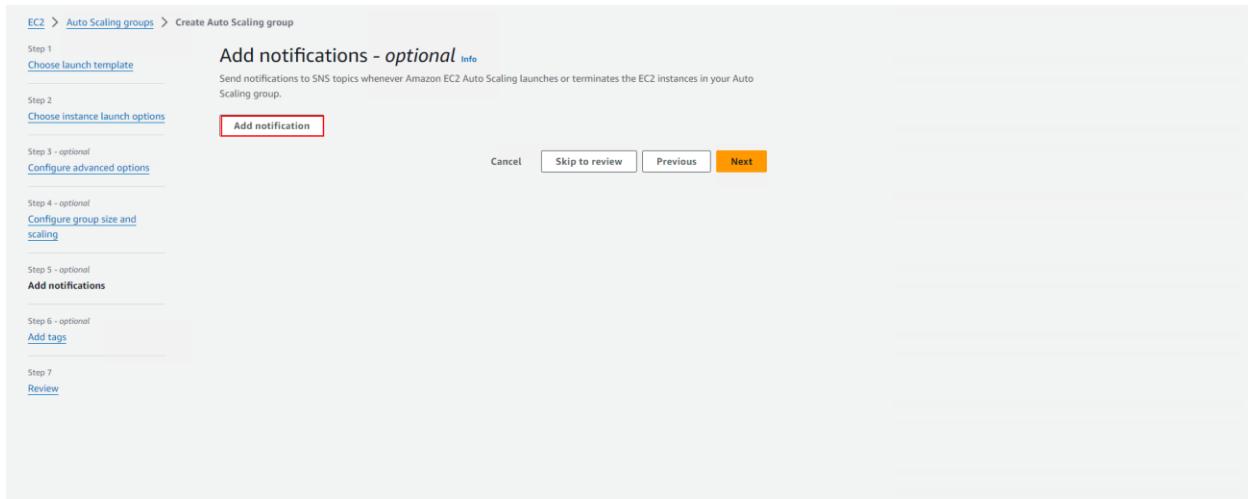
number of instances up to the **Maximum Value**, and if it drops to 50% or below, it will reduce the instances back to the **Desired Value**.



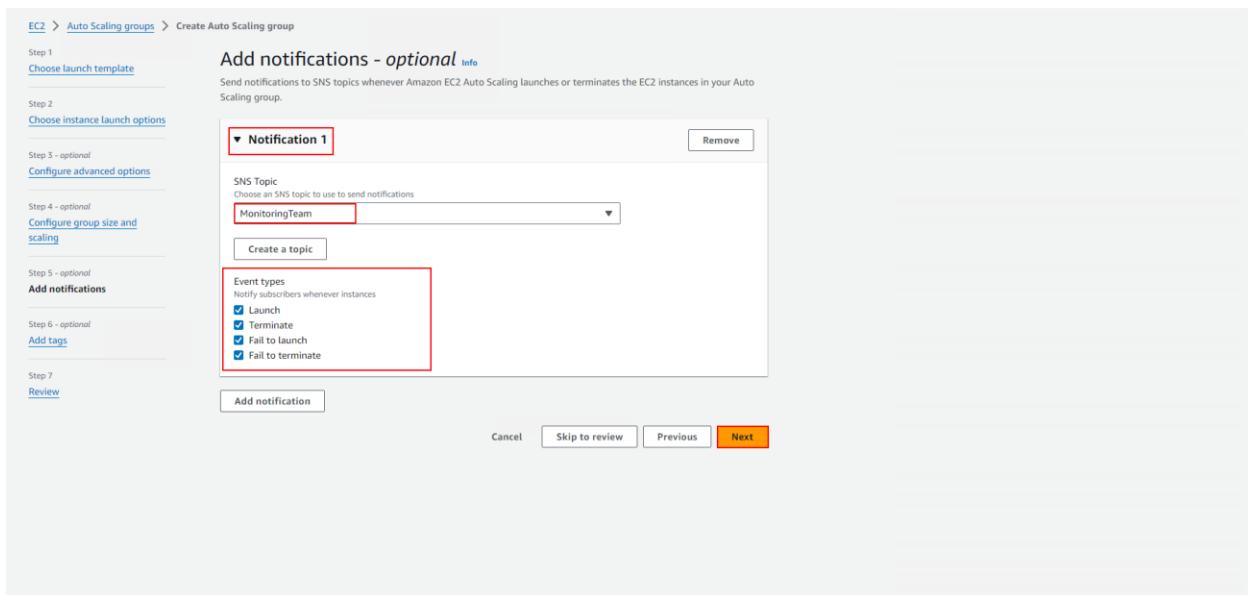
At this stage, click on the **Next** button.



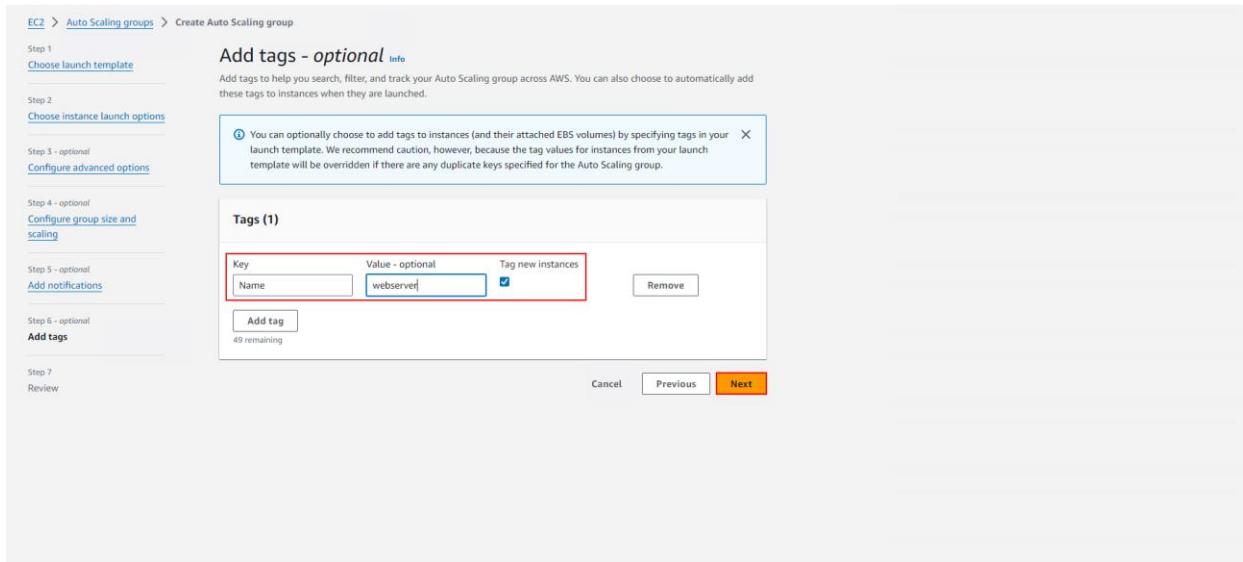
At this stage, click on the **Add Notification** button.



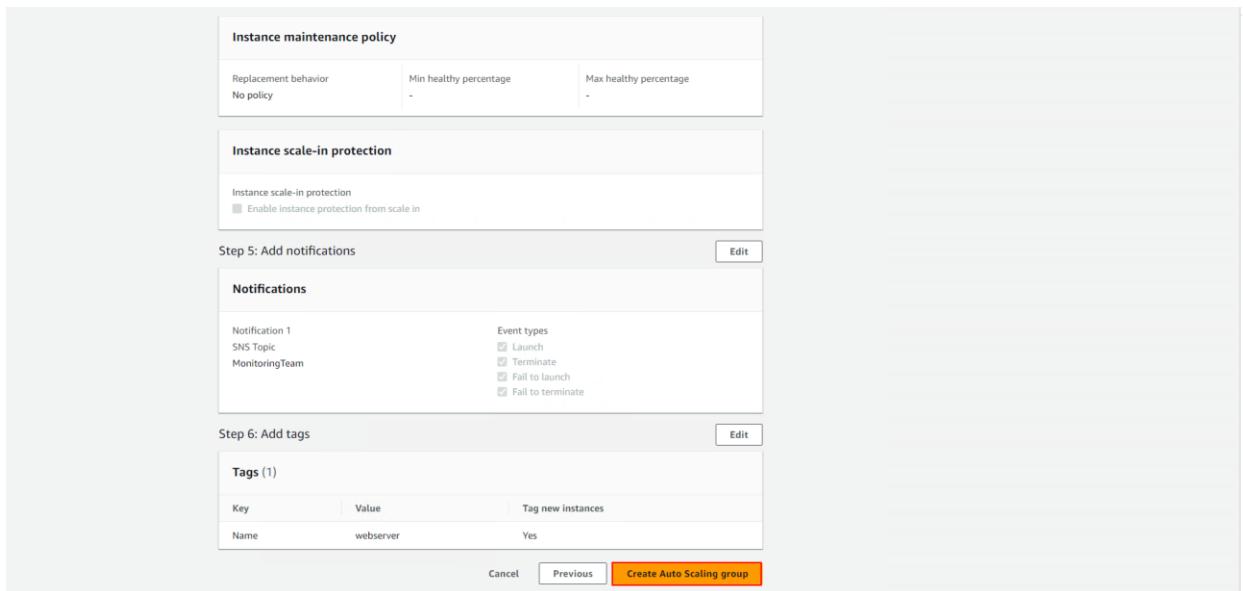
At this stage, select the **SNS Topic** and **Event Type**, then click on the **Next** button.



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.



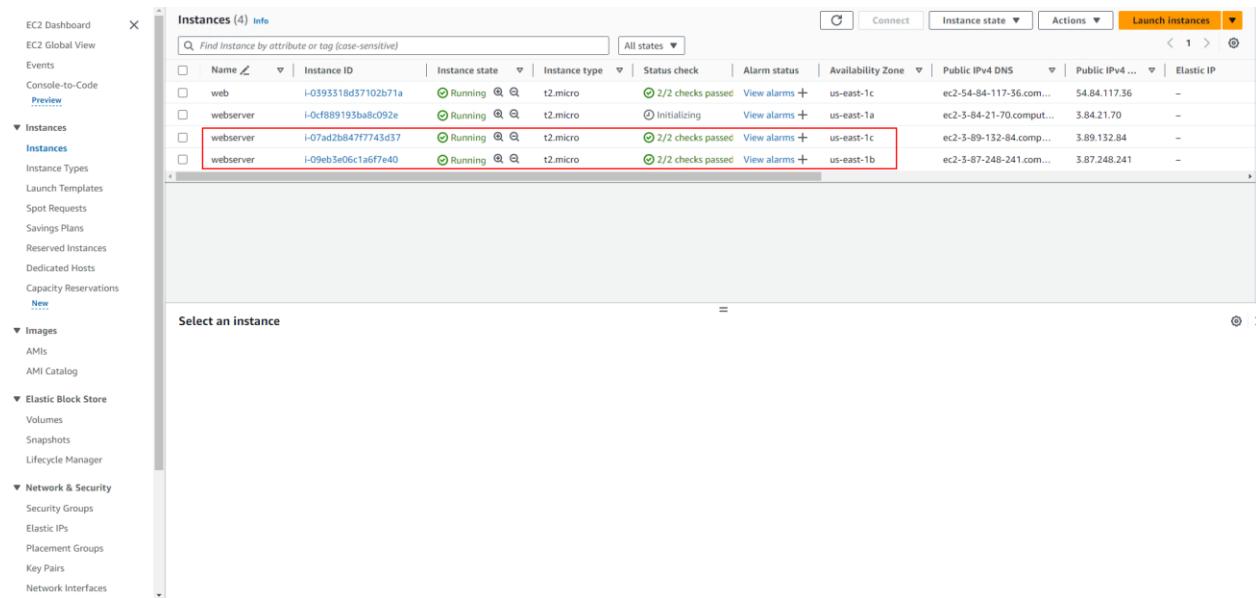
In the **Activity** section of the **Auto Scaling Group**, you can see that **two instances** have been created.

Status	Description	Cause	Start time	End time
Not yet in service	Launching a new EC2 instance: i-09eb3e06c1a6f7e40	At 2024-03-23T16:07:09Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 2. At 2024-03-23T16:07:20Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 2.	2024 March 23, 09:07:22 AM -07:00	
Not yet in service	Launching a new EC2 instance: i-07ad2b847f7743d37	At 2024-03-23T16:07:09Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 2. At 2024-03-23T16:07:20Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 2.	2024 March 23, 09:07:22 AM -07:00	

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

Instance ID	Lifecycle	Instance type	Weighted capacity	Launch template/...	Availability Zone	Health status	Protected from
i-07ad2b847f7743d37	InService	t2.micro	-	Health-Template Version	us-east-1c	Healthy	
i-09eb3e06c1a6f7e40	InService	t2.micro	-	Health-Template Version	us-east-1b	Healthy	

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



The screenshot shows the AWS EC2 Instances page with the following details:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
web	i-0393318d37102b71a	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1c	ec2-54-84-117-56.com...	54.84.117.36	-
webserver	i-0cf889193ba8c092e	Running	t2.micro	Initializing	View alarms +	us-east-1a	ec2-3-84-21-70.comput...	3.84.21.70	-
webserver	i-07ad2b847f7745d37	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1c	ec2-3-89-132-84.comp...	3.89.132.84	-
webserver	i-09eb3e06c1a6fe40	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-3-87-248-241.com...	3.87.248.241	-

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**.

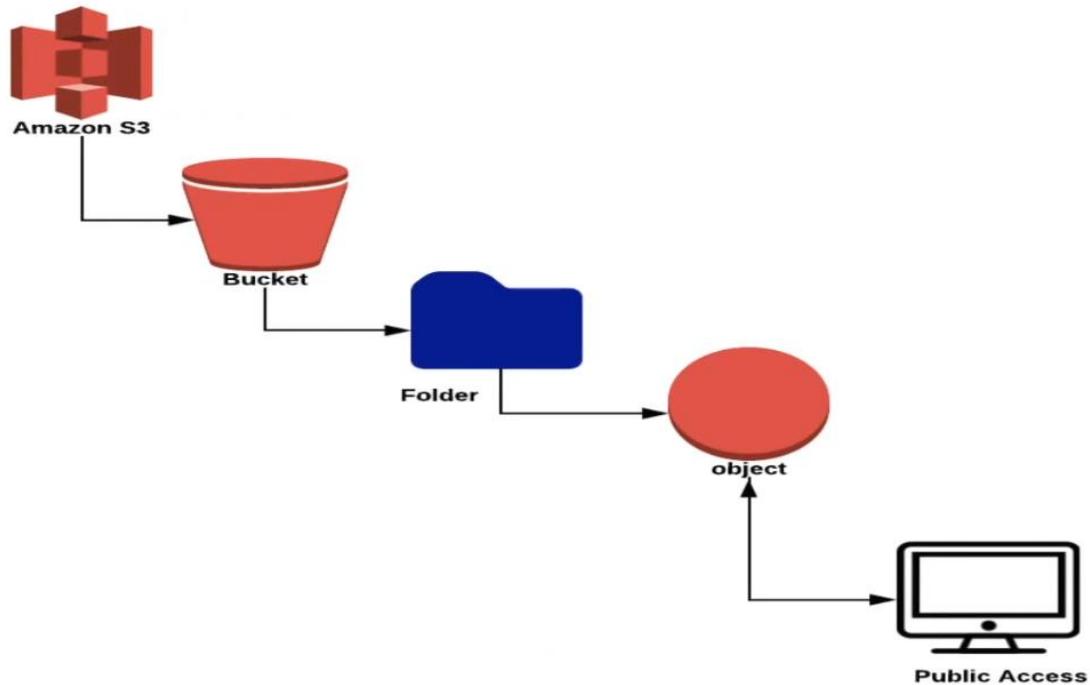
When you upload any data to your **S3 bucket**, the **bucket** acts as a **top-level storage** container—similar to a **folder** in S3.

Your data is **replicated across multiple S3 systems**, and there are **no limits** on storage.

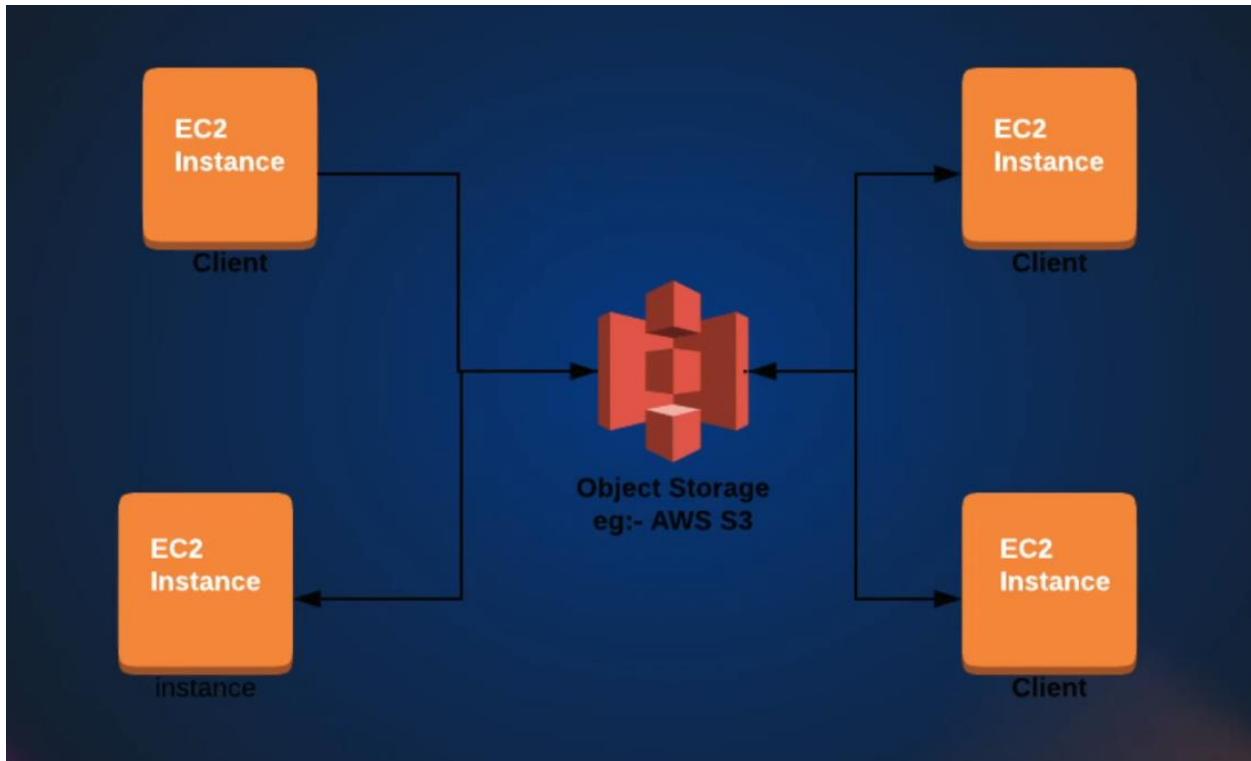
The **S3 service** offers **unlimited storage**, allowing you to store **as much data as you want**.

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:

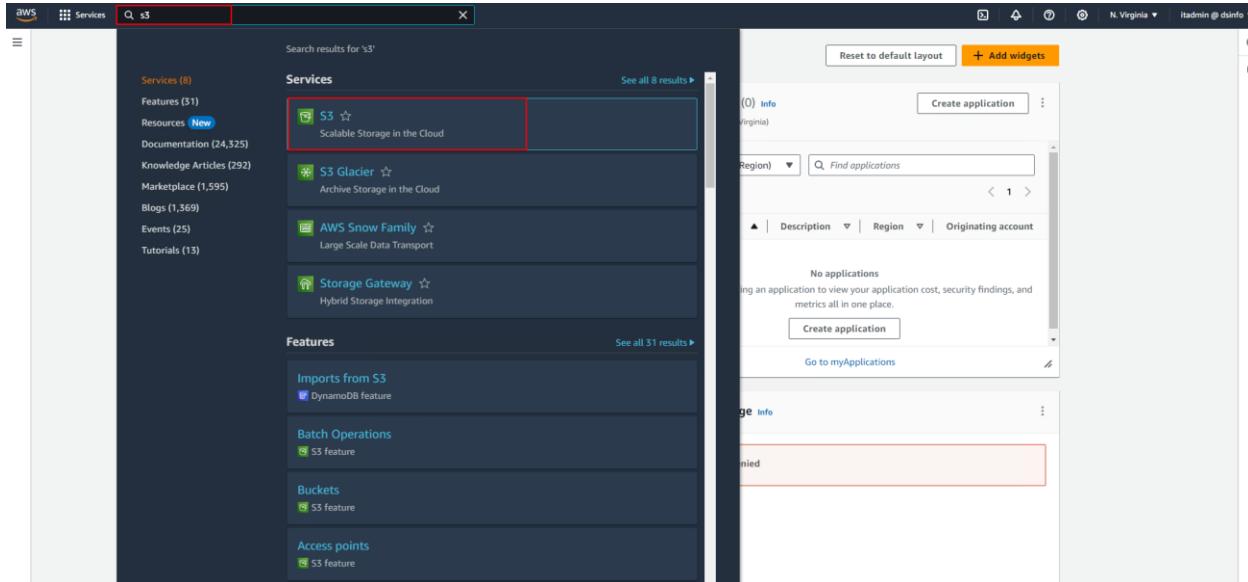
- Use **lifecycle policies** to move infrequently accessed data to lower-cost storage classes.

- 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

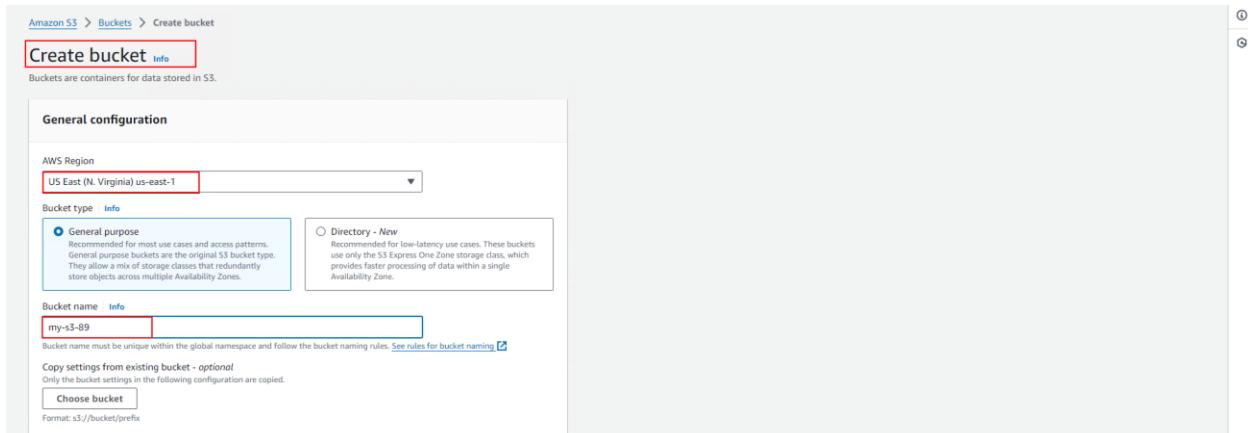
To use S3, type "S3" in the search bar and then click on S3 from the results.



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**.

The screenshot shows the "Block Public Access settings for this bucket" section with the "Block all public access" checkbox selected. It also shows the "Bucket Versioning" section where "Disable" is selected.

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

The screenshot shows the "Tags - optional (0)" section with a note about using tags to track storage costs and organize buckets. The "Default encryption info" section shows "Server-side encryption is automatically applied to new objects stored in this bucket." Under "Encryption type", "Server-side encryption with Amazon S3 managed keys (SSE-S3)" is selected. The "Bucket Key" section notes that using an SSE-KMS Bucket Key reduces encryption costs by lowering calls to AWS KMS. Bucket Keys aren't supported for SSE-KMS. The "Advanced settings" section is expanded, showing a note about uploading files after creation. At the bottom are "Cancel" and "Create bucket" buttons.

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

The screenshot shows the AWS S3 Buckets page. At the top, a green banner indicates "Successfully created bucket 'my-s3-89'". Below the banner, there's an "Account snapshot" section with a "Storage lens provides visibility into storage usage and activity trends" message and a "View Storage Lens dashboard" button. The main area is titled "General purpose buckets (1) Info". It shows a single bucket named "my-s3-89" with the following details:

Name	AWS Region	Access	Creation date
my-s3-89	US East (N. Virginia) us-east-1	Bucket and objects not public	March 24, 2024, 23:34:36 (UTC-07:00)

Actions available for the bucket include "Copy ARN", "Empty", "Delete", and "Create bucket".

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

The screenshot shows the AWS S3 Objects page for the "my-s3-89" bucket. The top navigation bar includes "Amazon S3 > Buckets > my-s3-89" and tabs for "Objects", "Properties", "Permissions", "Metrics", "Management", and "Access Points". The "Objects (0) Info" section states: "Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more". A search bar "Find objects by prefix" is present. The main table header includes columns for "Name", "Type", "Last modified", "Size", and "Storage class". A message "No objects" and "You don't have any objects in this bucket." is displayed. An "Upload" button is located at the bottom of the table.

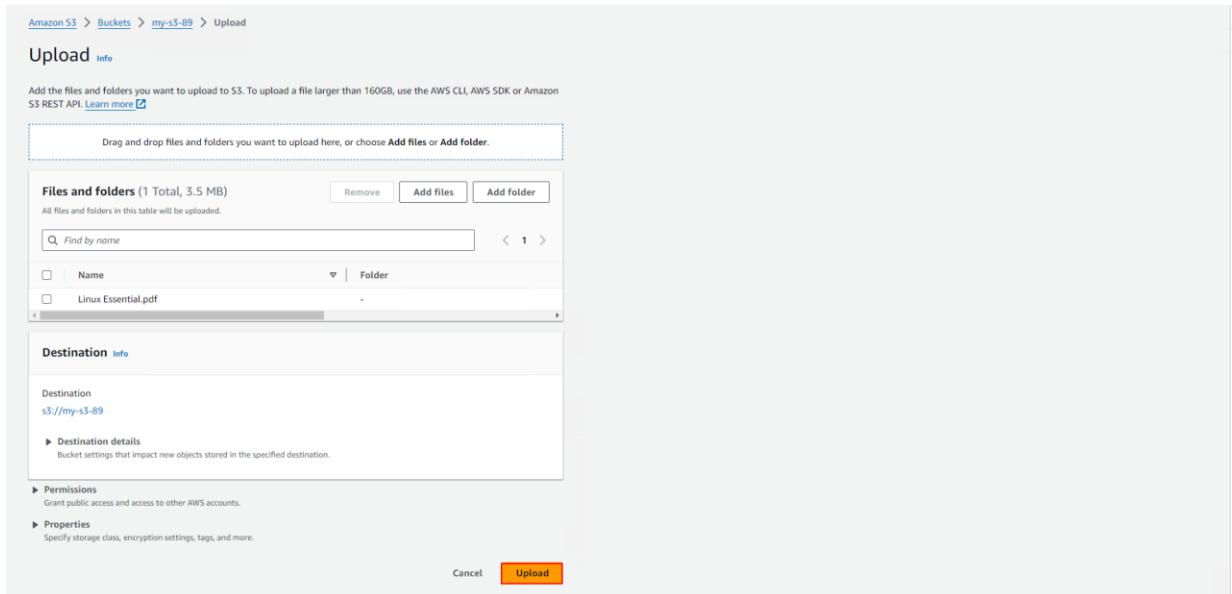
At this stage, click on the **Add Files** button.

The screenshot shows the AWS S3 'Upload' interface. At the top, it says 'Amazon S3 > Buckets > my-s3-89 > Upload'. Below that is a 'Upload' section with a note about file size limits. A large dashed box area is labeled 'Drag and drop files and folders you want to upload here, or choose Add files or Add folder.' To its right are 'Remove', 'Add files' (which is highlighted in red), and 'Add folder' buttons. Below this is a table titled 'Files and folders (0)' with a search bar and a 'Find by name' input field. The table has columns for 'Name' and 'Folder'. A message at the bottom of the table says 'No files or folders' and 'You have not chosen any files or folders to upload.' On the left, there's a 'Destination' section with a dropdown set to 's3://my-s3-89' and a 'Destination details' link. The entire interface is framed by a light gray border.

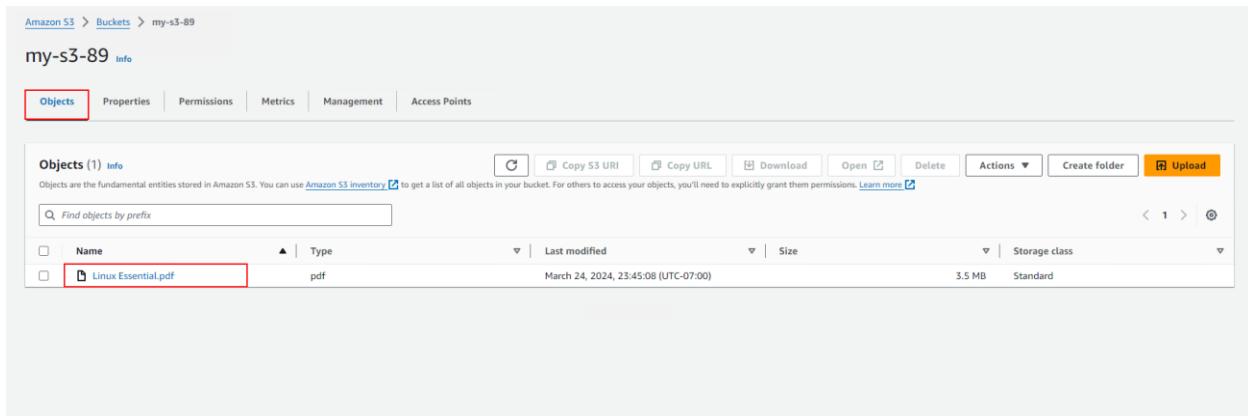
Select your desired file from your computer, then click the **Open** button.

This screenshot is similar to the previous one, showing the AWS S3 'Upload' interface. However, a Windows 'Open' file dialog box is overlaid on the right side of the screen. The dialog box shows a list of items on the desktop, including 'Administrator', 'This PC', 'Network', and 'Libraries'. A file named 'Linux Essential' is selected, indicated by a red rectangular highlight around its icon and name. The 'File name' field at the bottom contains 'Linux Essential' and the 'Open' button is highlighted in red. The background S3 interface remains visible through the dialog box.

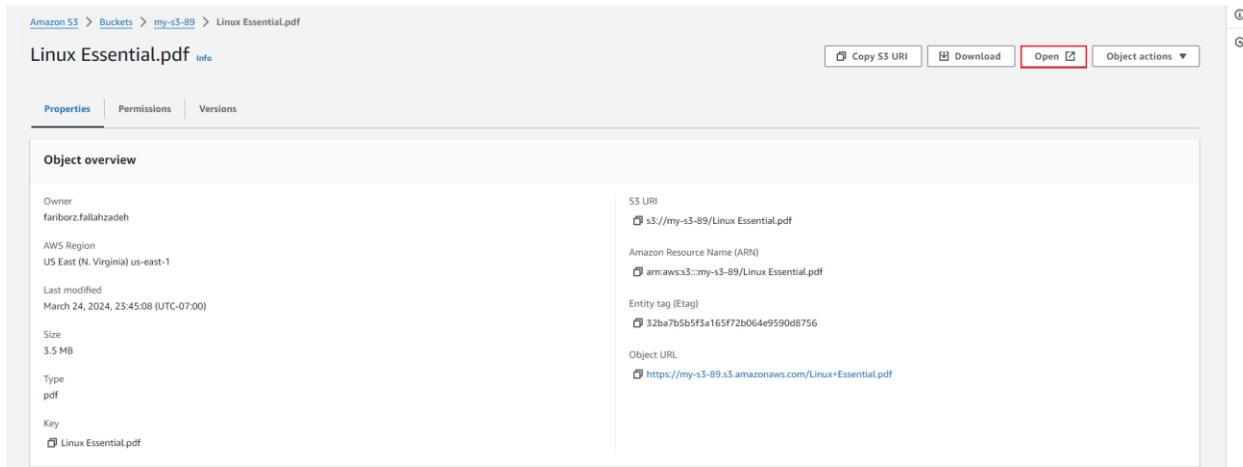
Then, click on the **Upload** button.



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.



The screenshot shows the 'Object overview' section of the Amazon S3 console. At the top right, there are buttons for 'Copy S3 URI', 'Download', 'Open' (which is highlighted with a red box), and 'Object actions'. Below the buttons, there are tabs for 'Properties', 'Permissions', and 'Versions', with 'Properties' being the active tab. The 'Object overview' table contains the following data:

Attribute	Value
Owner	fariborz.fallahzadeh
AWS Region	US East (N. Virginia) us-east-1
Last modified	March 24, 2024, 23:45:08 (UTC-07:00)
Size	3.5 MB
Type	pdf
Key	Linux Essential.pdf
S3 URI	s3://my-s3-89/Linux Essential.pdf
Amazon Resource Name (ARN)	arn:aws:s3:::my-s3-89/Linux Essential.pdf
Entity tag (Etag)	32ba7b5b5f5a165f72b064e9590d8756
Object URL	https://my-s3-89.s3.amazonaws.com/Linux+Essential.pdf

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.

The screenshot shows the 'Object overview' section of the Amazon S3 console. The object details are as follows:

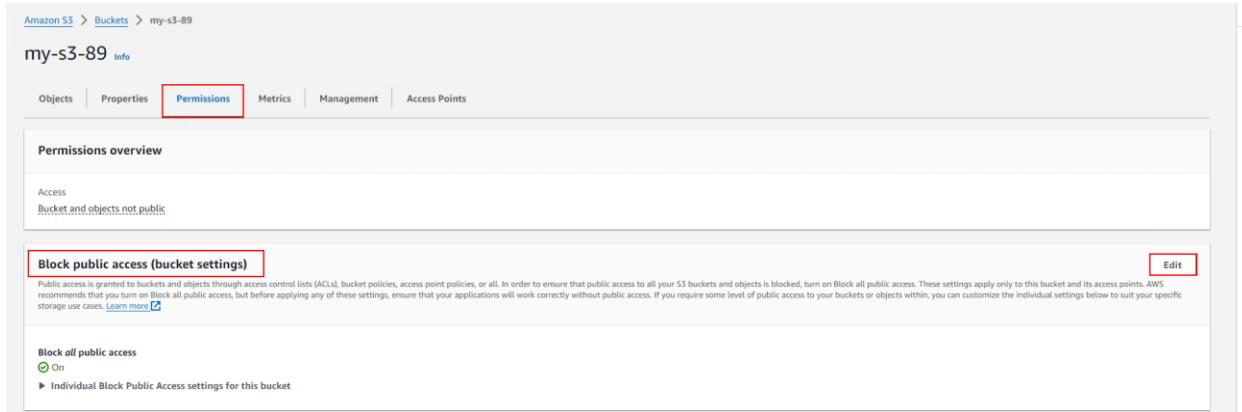
Key	Value
Owner	fariborz.fallahzadeh
AWS Region	US East (N. Virginia) us-east-1
Last modified	March 24, 2024, 23:45:08 (UTC-07:00)
Size	3.5 MB
Type	pdf
S3 URI	s3://my-s3-89/Linux+Essential.pdf
Amazon Resource Name (ARN)	arn:aws:s3:::my-s3-89/Linux+Essential.pdf
Entity tag (ETag)	32ba7b5b5f3a165f72b064e9590d8756
Object URL	https://my-s3-89.s3.amazonaws.com/Linux+Essential.pdf

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

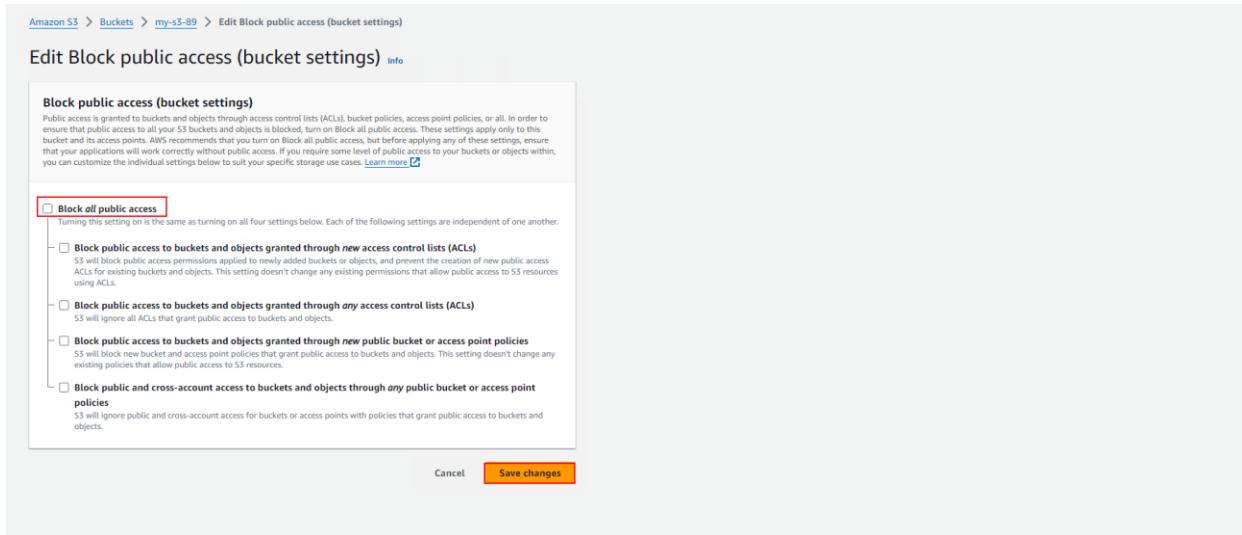
The browser window displays a 403 Access Denied error page. The URL in the address bar is <https://my-s3-89.s3.amazonaws.com/Linux+Essential.pdf>. The error message is as follows:

```
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestId>03PT8R68TQTP6F</RequestId>
<HostId>k8RHTXj1YKCDSHkj1Hw/hCUq0y95d014UE3N6EnpvLGLD2p+VnTrB514LOYYv58jje/dE5s=</HostId>
</Error>
```

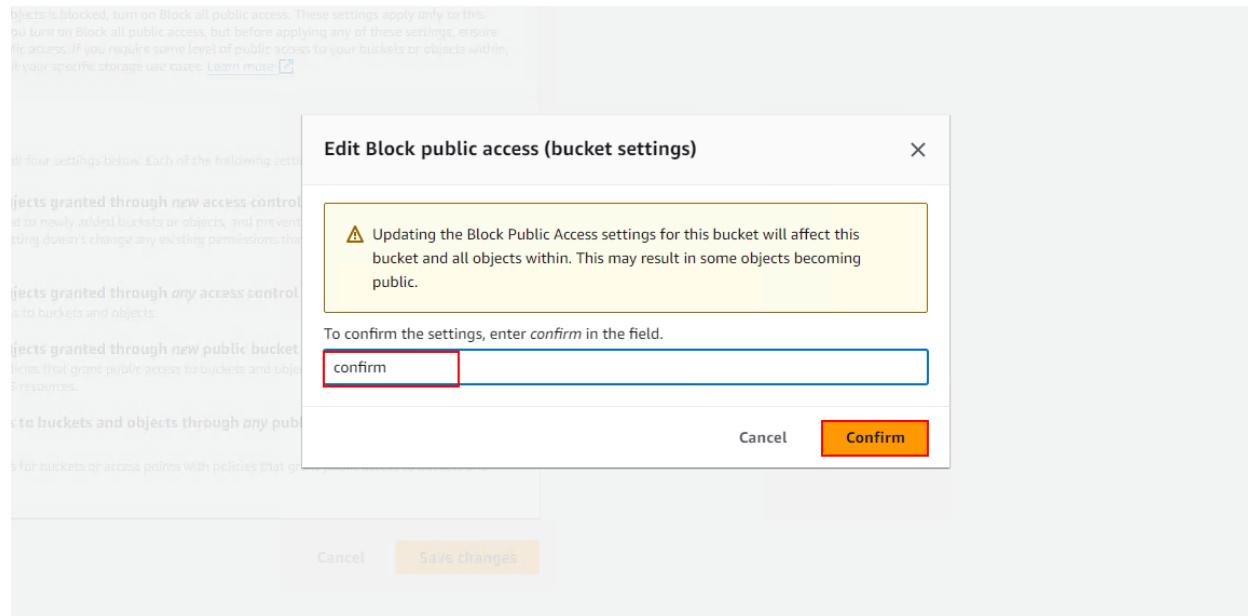
To enable **public access** to the file, go to the **Permissions** tab of the selected file, then click on the **Edit** button under the **Block Public Access** section.



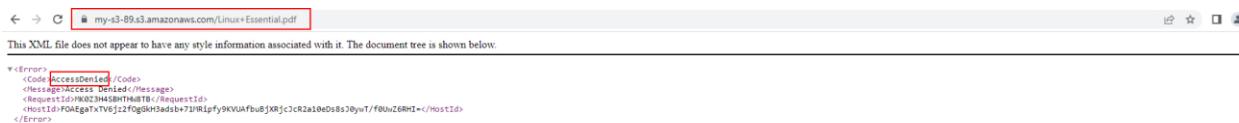
At this stage, **uncheck** the option **Block all public access**, and then click on the **Save Changes** button.



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**.

201

Instructor: Fariborz Fallahzadeh

Email Address:fariborz.fallahzadeh@gmail.com

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

Amazon S3 > Buckets > my-s3-89 > LinuxEssential.pdf

LinuxEssential.pdf [Info](#)

[Properties](#) [Permissions](#) [Versions](#)

Object overview

Owner
fariborz.fallahzadeh

AWS Region
US East (N. Virginia) us-east-1

Last modified
March 25, 2024, 00:00:02 (UTC-07:00)

Size
3.5 MB

Type
pdf

Key
LinuxEssential.pdf

S3 URI
<s3://my-s3-89/LinuxEssential.pdf>

Amazon Resource Name (ARN)
<arn:aws:s3:::my-s3-89/LinuxEssential.pdf>

Entity tag (Etag)
<32ba7b5b5f3a165f72b064e9590d8756>

Object URL
<https://my-s3-89.s3.amazonaws.com/LinuxEssential.pdf>

Object actions

- Copy S3 URI
- Download
- Open
- Object actions ▾
- Download as
- Share with a presigned URL
- Calculate total size
- Copy
- Move
- Initiate restore
- Query with S3 Select
- Edit actions
- Rename object
- Edit storage class
- Edit server-side encryption
- Edit metadata
- Edit tags
- Make public using ACL**

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

Amazon S3 > Buckets > my-s3-89 > Make public

Make public [Info](#)

The make public action enables public read access in the object access control list (ACL) settings. [Learn more](#)

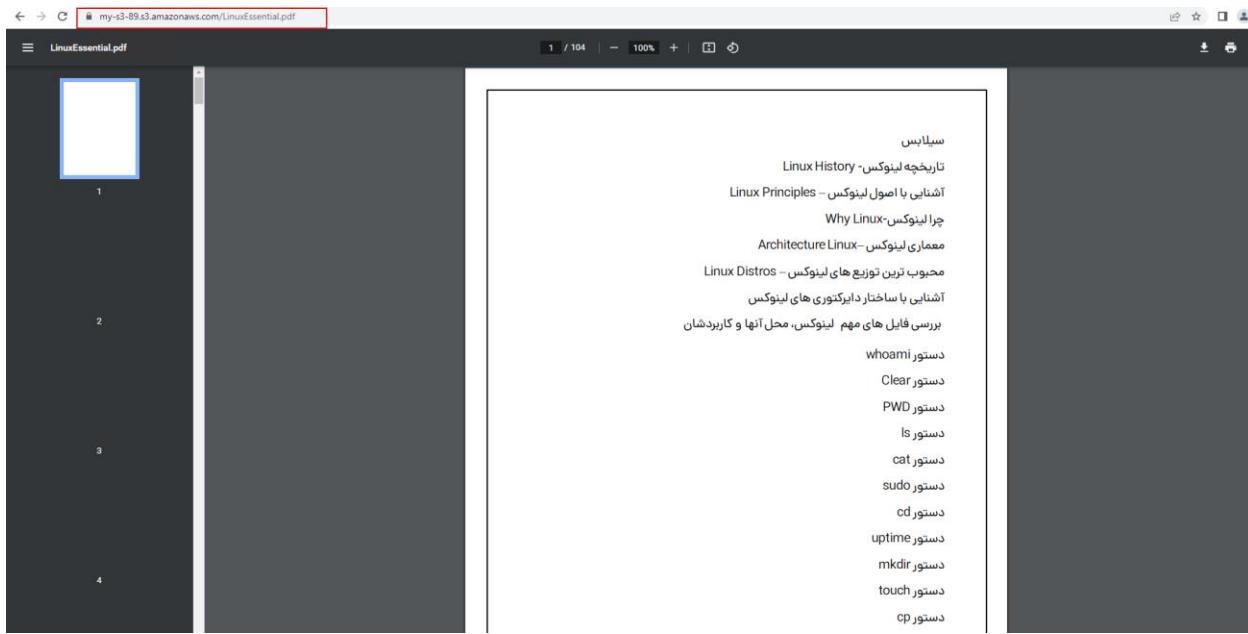
⚠ When public read access is enabled and not blocked by Block Public Access settings, anyone in the world can access the specified objects.

Specified objects

Name	Type	Last modified	Size
LinuxEssential.pdf	pdf	March 25, 2024, 00:00:02 (UTC-07:00)	3.5 MB

[Cancel](#) **Make public**

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

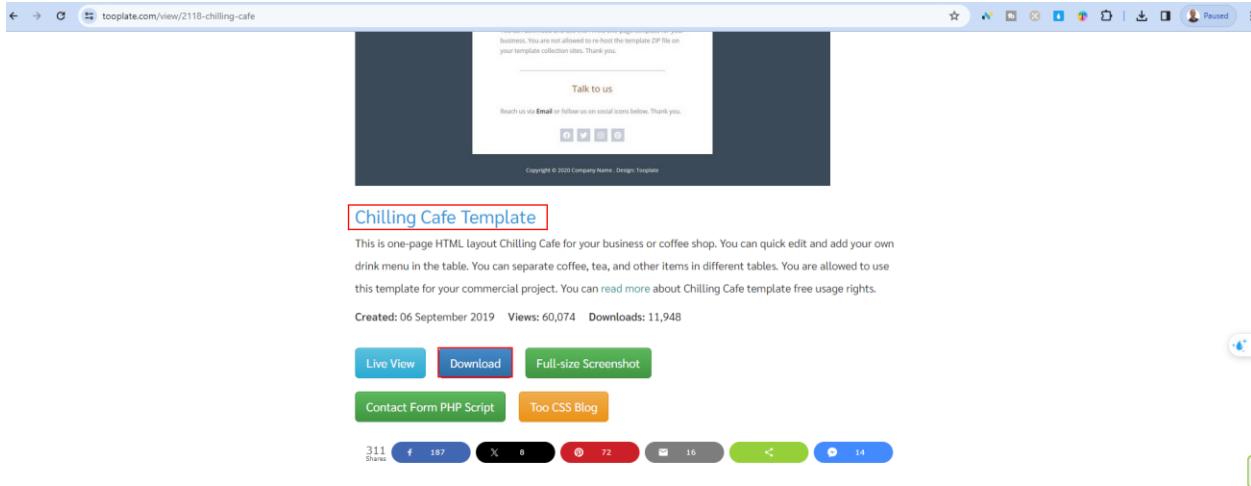


How to Host a Static Website Using an S3 Bucket

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.

The screenshot shows the 'Objects (1) Info' section of the S3 bucket 'my-s3-89'. A single file, 'LinuxEssential.pdf', is listed. The file is a PDF document, last modified on March 25, 2024, at 00:00:02 (UTC-07:00), and is 3.5 MB in size. The storage class is Standard. At the top right, there is a prominent orange 'Upload' button.

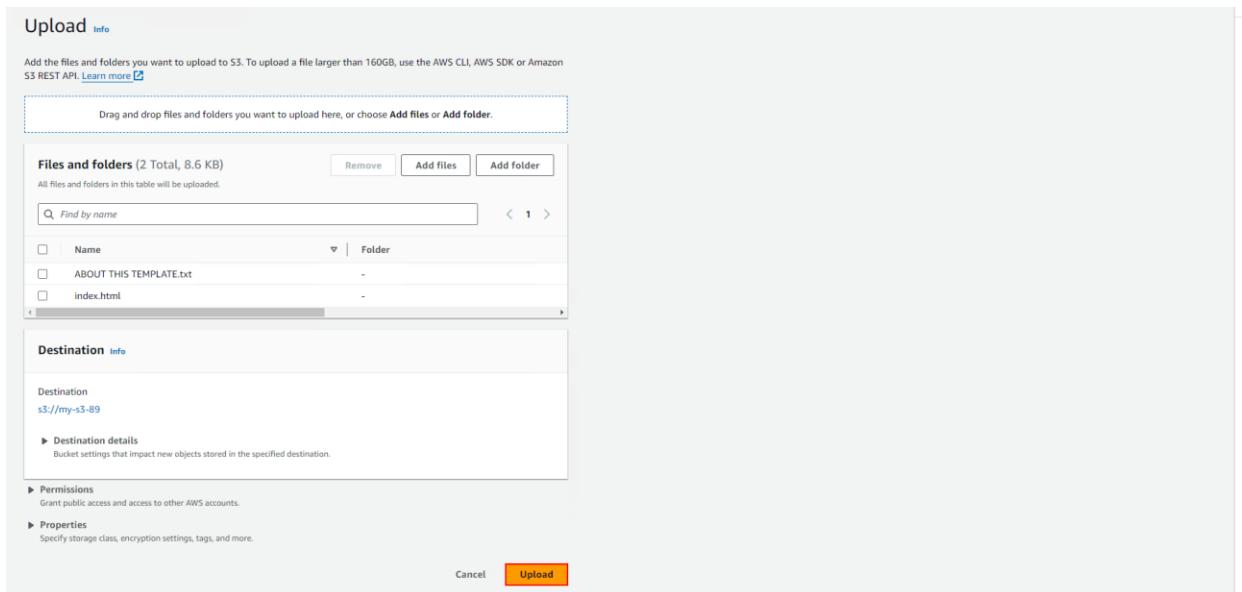
At this stage, click on the **Add Files** button.

The screenshot shows the 'Upload' step in the S3 console. It displays a 'Files and folders (0)' list with a red box around the 'Add files' button. Below the list, it says 'No files or folders' and 'You have not chosen any files or folders to upload.'

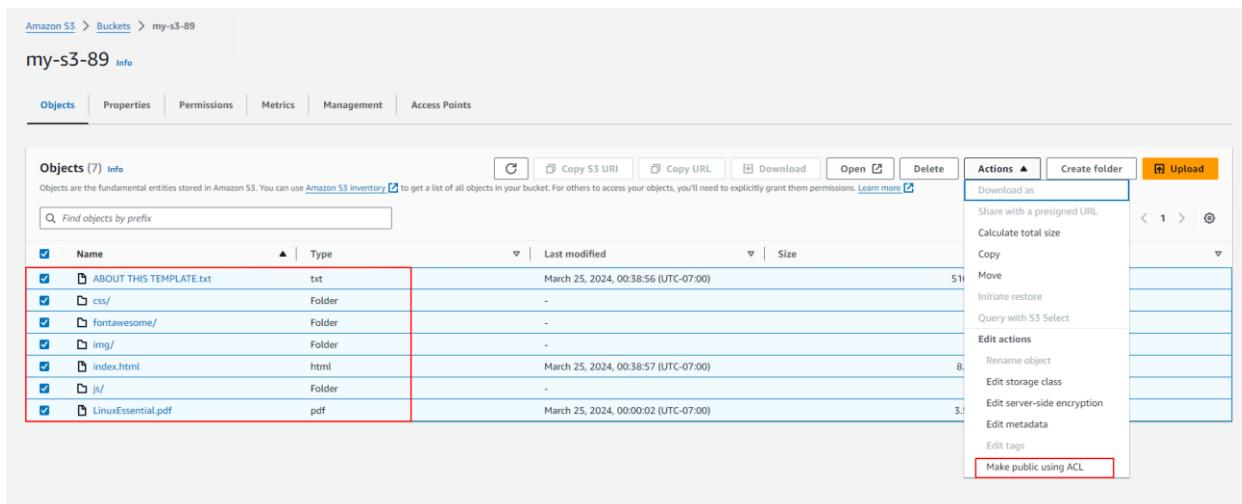
Then, select the files and click the **Open** button.

The screenshot shows the 'Open' file dialog box. It lists several files and folders: 'css', 'fontawesome', 'img', 'js', 'ABOUT THIS TEMPLATE', and 'index'. The 'index' file is selected, indicated by a red box. The 'File name:' dropdown shows 'ABOUT THIS TEMPLATE' 'index'. At the bottom right of the dialog, there is an 'Open' button with a red box around it.

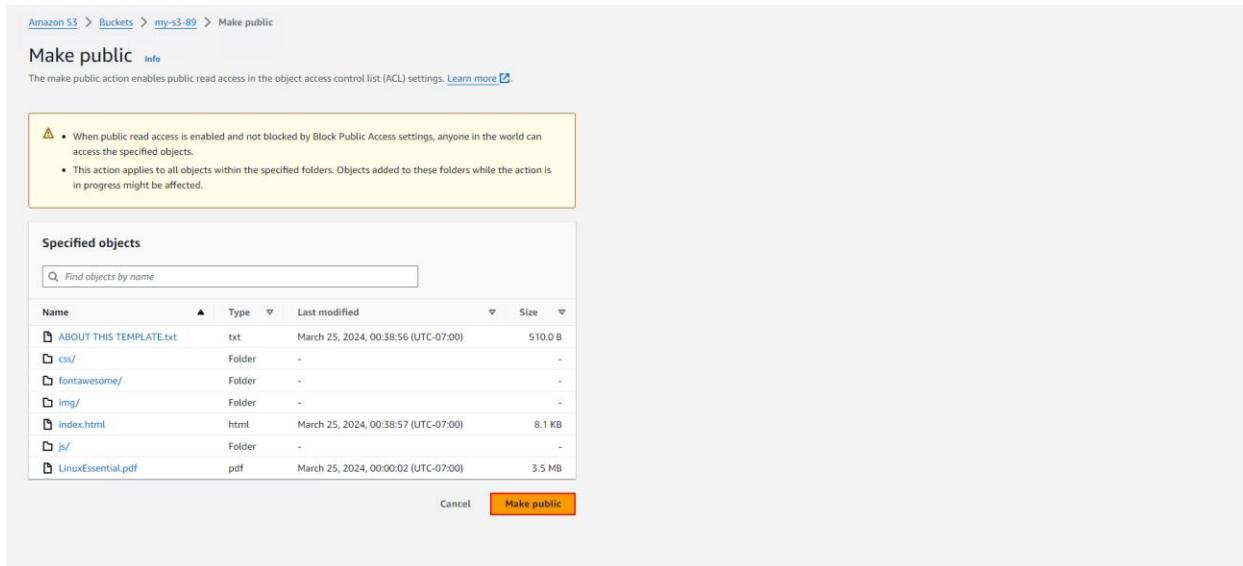
Then, click on the **Upload** button.



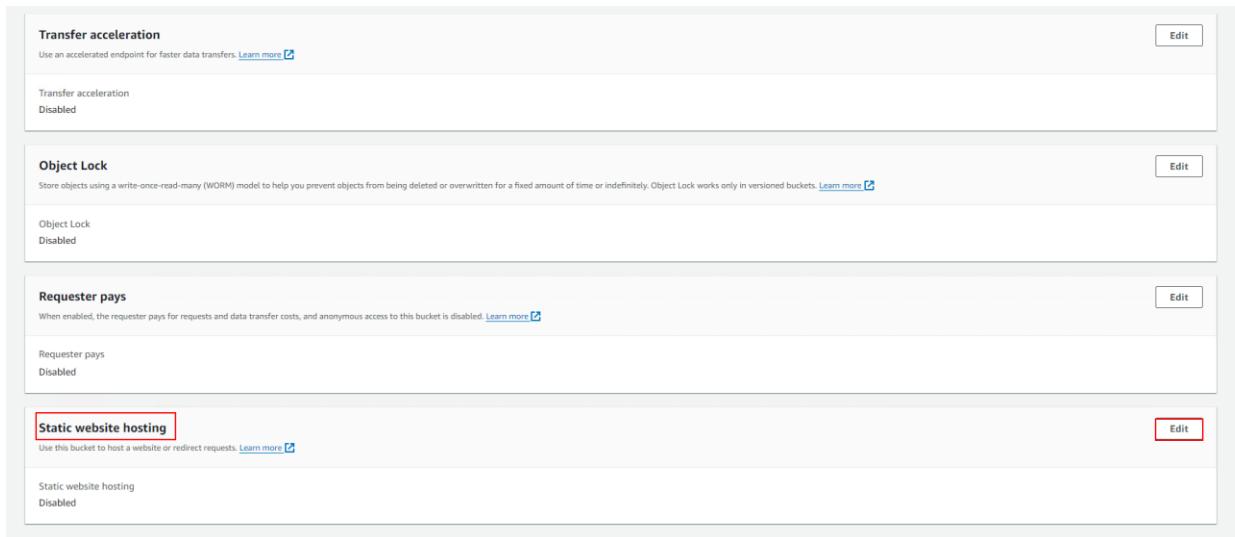
At this stage, select all the **template files**, then from the **Actions** menu, choose **Make Public Using ACL**.



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

The screenshot shows the 'Edit static website hosting' configuration page for a bucket named 'my-s3-89'. The 'Static website hosting' section is active, with 'Enable' selected. Under 'Host type', 'Host a static website' is selected. A note about making content publicly readable is visible. The 'Index document' field contains 'index.html', and the 'Error document - optional' field contains 'error.html'. A large red box highlights the 'Index document' and 'Error document' fields.

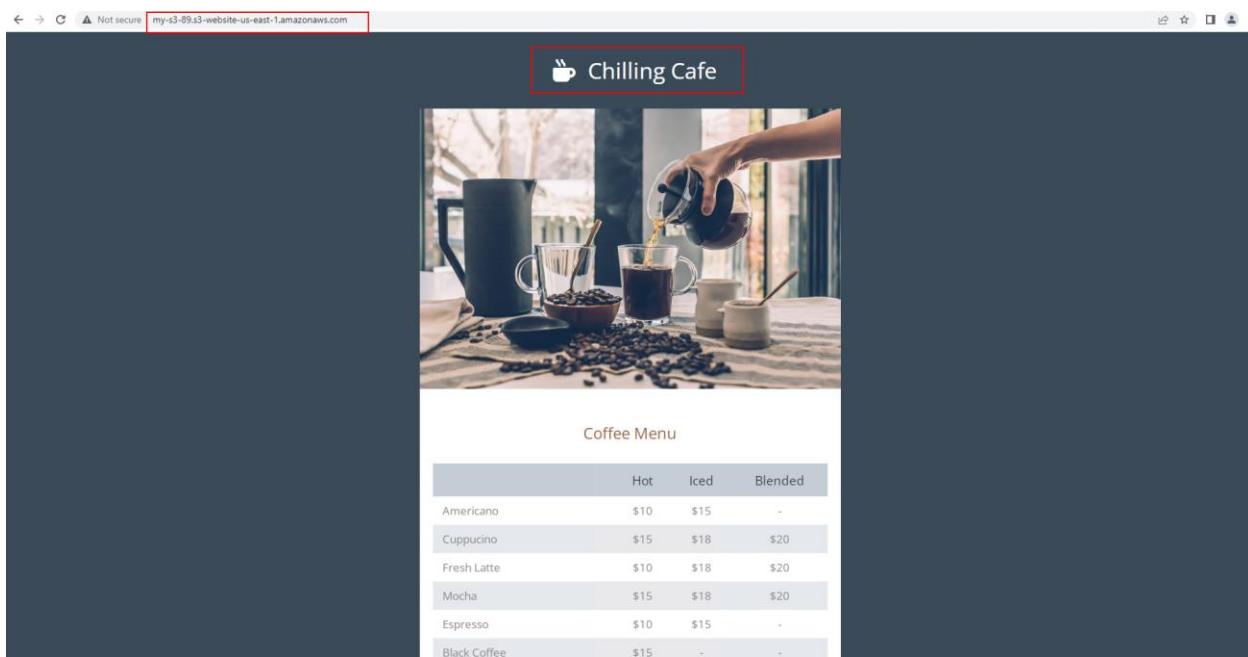
And finally, click on the **Save Changes** button.

The screenshot shows the 'Redirection rules' configuration dialog box. It displays a JSON editor with one rule defined. The rule has the ID '1' and a single condition. The bottom status bar shows 'JSON' and 'Ln 1, Col 1'. The 'Save changes' button is highlighted with a red box.

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

The screenshot shows the 'Static website hosting' section of the AWS S3 Bucket Properties page. It includes fields for 'Static website hosting' (Enabled), 'Hosting type' (Bucket hosting), and 'Bucket website endpoint' (http://my-s3-89.s3-website-us-east-1.amazonaws.com). A red box highlights the endpoint URL.

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

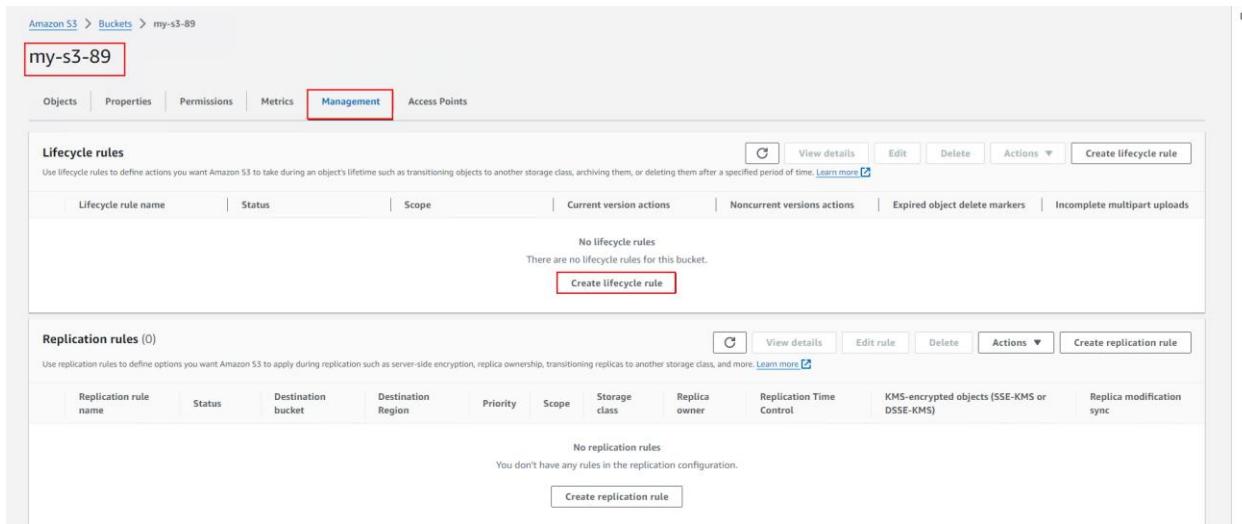


How to Use S3 Bucket Lifecycle

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.



The screenshot shows the Amazon S3 Management console for the bucket 'my-s3-89'. The 'Management' tab is selected. In the 'Lifecycle rules' section, there is a table with columns: Lifecycle rule name, Status, Scope, Current version actions, Noncurrent versions actions, Expired object delete markers, and Incomplete multipart uploads. A message states 'No lifecycle rules' and 'There are no lifecycle rules for this bucket.' A 'Create lifecycle rule' button is highlighted with a red box. Below this, the 'Replication rules (0)' section is shown, also with a 'Create replication rule' button highlighted with a red box.

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

The screenshot shows the 'Create lifecycle rule' configuration page in the Amazon S3 console. The URL in the address bar is: [Amazon S3 > Buckets > my-s3-89 > Lifecycle configuration > Create lifecycle rule](#). The page title is 'Create lifecycle rule' with a 'Info' link. The main section is 'Lifecycle rule configuration'. Under 'Lifecycle rule name', the value 'archive-polices' is entered in a text input field, which is highlighted with a red border. Below it, a note says 'Up to 255 characters'. Under 'Choose a rule scope', the radio button 'Limit the scope of this rule using one or more filters' is selected. Under 'Filter type', it says 'You can filter objects by prefix, object tags, object size, or whatever combination suits your usecase.' A 'Prefix' section shows a text input field with 'log' entered, also highlighted with a red border. A note below says 'Don't include the bucket name in the prefix. Using certain characters in key names can cause problems with some applications and protocols.' with a 'Learn more' link. Under 'Object tags', there is a note 'You can limit the scope of this rule to the key/value pairs added below.' and a 'Add tag' button.

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.

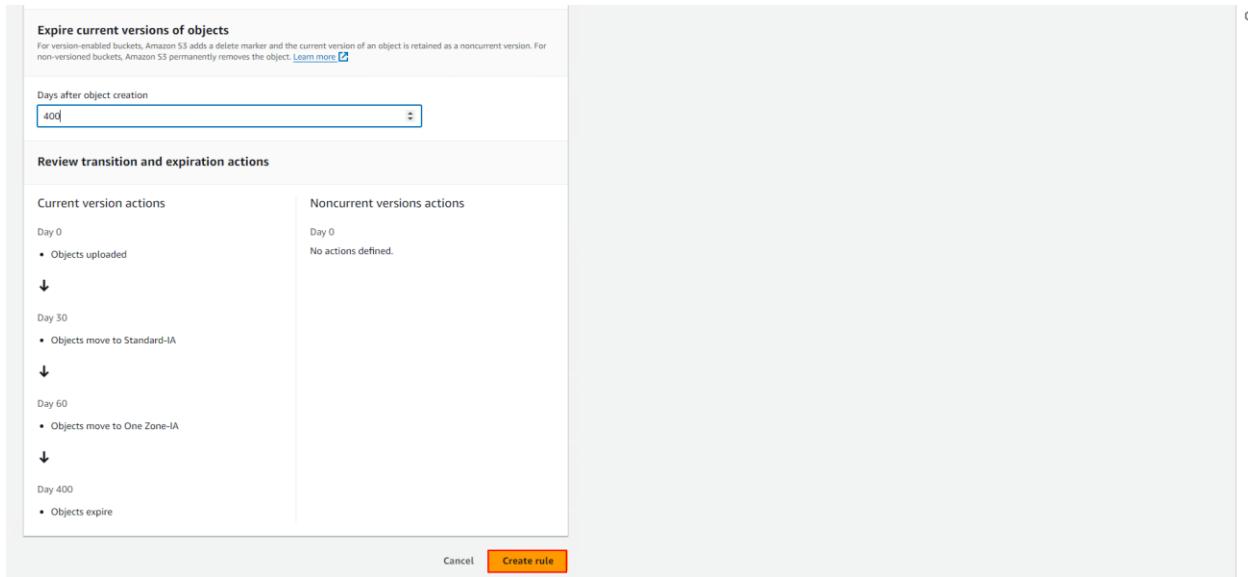
In this step, you must define how many **days** the data should remain in each **storage class** before it should **transition** to the next storage class.

The screenshot shows the 'Transition current versions of objects between storage classes' section. It includes a table for defining storage class transitions:

Choose storage class transitions	Days after object creation
Standard-IA	30
One Zone-IA	60

Below this, the 'Expire current versions of objects' section is shown, featuring a dropdown for 'Days after object creation' set to 400. The 'Review transition and expiration actions' section at the bottom shows 'Day 0' for both current and noncurrent version actions.

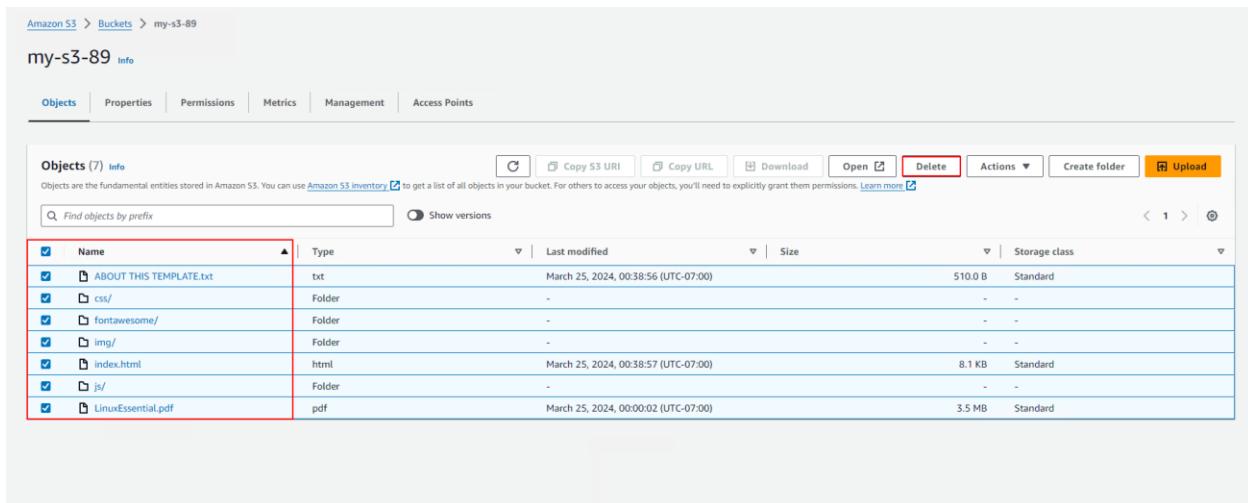
And finally, click on the **Create Rule** button.



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

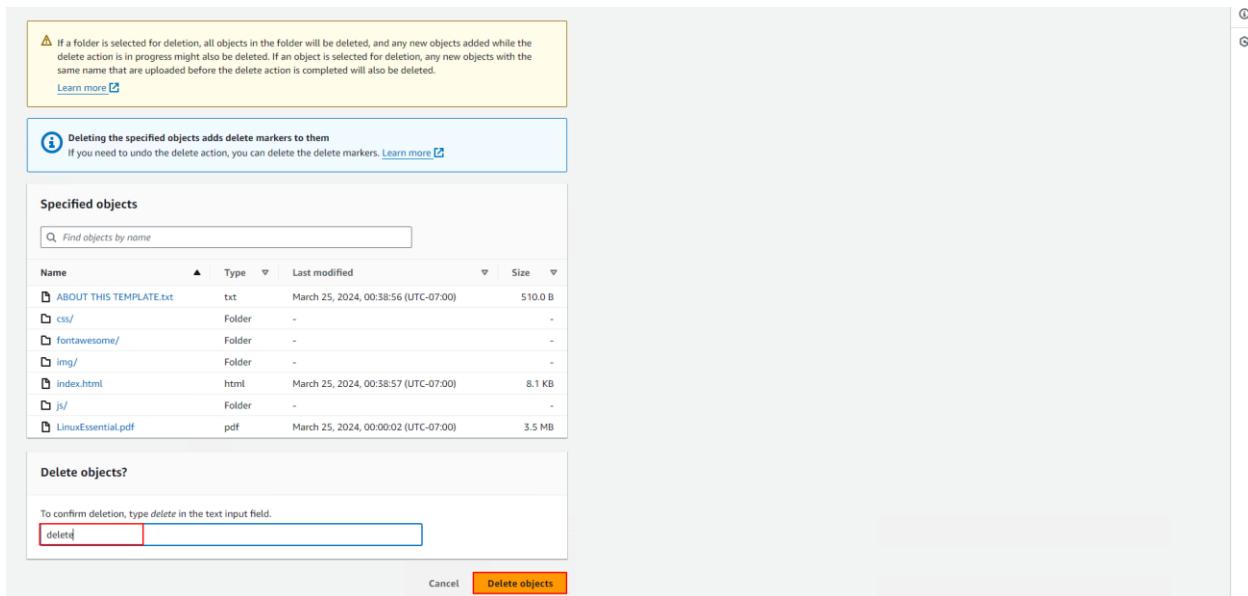
The screenshot shows the 'Lifecycle configuration' page in the AWS S3 console. The URL is 'Amazon S3 > Buckets > my-s3-89 > Lifecycle configuration'. The title 'Lifecycle configuration' is highlighted with a red box. Below it, a note says: 'To manage your objects so that they are stored cost effectively throughout their lifecycle, configure their lifecycle. A lifecycle configuration is a set of rules that define actions that Amazon S3 applies to a group of objects. Lifecycle rules run once per day.' A note about 'Learn more' is also present. The main area shows 'Lifecycle rules (1)'. A search bar 'Find lifecycle rules by name' is at the top. The table below has columns: 'Lifecycle rule name', 'Status', 'Scope', 'Current version actions', 'Noncurrent versions actions', 'Expired object delete mar...', and 'Incomplete multipart up...'. The first row in the table is highlighted with a red box and labeled 'archive-polices'. The status is 'Enabled' and the scope is 'Filtered'. The 'Current version actions' is 'Transition to Standard-IA, then One Zone-IA'. The 'Actions' bar at the top right includes 'View details', 'Edit', 'Delete', 'Actions', and 'Create lifecycle rule'. Navigation arrows and a refresh icon are also present.

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



The screenshot shows the Amazon S3 'Objects' page for the bucket 'my-s3-89'. At the top, there are tabs for Objects, Properties, Permissions, Metrics, Management, and Access Points. Below the tabs is a toolbar with actions like Copy S3 URI, Copy URL, Download, Open, Delete (which is highlighted with a red box), Actions, Create folder, and Upload. A search bar and a 'Show versions' link are also present. The main area displays a table of objects with columns for Name, Type, Last modified, Size, and Storage class. Several objects are selected, indicated by a checked checkbox in the first column. The objects listed include 'ABOUT THIS TEMPLATE.txt' (txt, March 25, 2024, 510.0 B, Standard), 'css/' (Folder, -), 'fontawesome/' (Folder, -), 'img/' (Folder, -), 'index.html' (html, March 25, 2024, 8.1 KB, Standard), 'js/' (Folder, -), and 'LinuxEssential.pdf' (pdf, March 25, 2024, 3.5 MB, Standard).

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



This screenshot shows a modal dialog box titled 'Delete objects?' with a yellow warning banner at the top. The banner states: 'If a folder is selected for deletion, all objects in the folder will be deleted, and any new objects added while the delete action is in progress might also be deleted. If an object is selected for deletion, any new objects with the same name that are uploaded before the delete action is completed will also be deleted.' Below the banner is an information message: 'Deleting the specified objects adds delete markers to them. If you need to undo the delete action, you can delete the delete markers.' The 'Specified objects' section lists the same set of files as the previous screenshot. Below this is a text input field containing the word 'delete'. At the bottom right of the dialog is a large orange 'Delete objects' button, which is also highlighted with a red box.

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

The screenshot shows the 'Buckets' section of the Amazon S3 console. At the top, there's an 'Account snapshot' box with a 'View Storage Lens dashboard' button. Below it, tabs for 'General purpose buckets' and 'Directory buckets' are shown, with 'General purpose buckets' selected. A search bar labeled 'Find buckets by name' is present. A table lists one bucket: 'my-s3-89' (selected), located in 'US East (N. Virginia) us-east-1'. The table includes columns for Name, AWS Region, Access, and Creation date. Buttons for 'Copy ARN', 'Empty', and 'Delete' are at the top right of the table, with 'Delete' highlighted with a red border. Navigation arrows and a refresh icon are at the bottom right of the table area.

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

The screenshot shows the 'Empty bucket' confirmation dialog for the 'my-s3-89' bucket. At the top, the path 'Amazon S3 > Buckets > my-s3-89 > Empty bucket' is visible. The main area contains a warning message about deleting all objects and a note about lifecycle rules. Below this, a section asks 'Permanently delete all objects in bucket "my-s3-89"?'. It includes a text input field with 'permanently delete' typed into it and a 'Go to lifecycle rule configuration' link. At the bottom, there are 'Cancel' and 'Empty' buttons, with 'Empty' highlighted with a red border.

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.

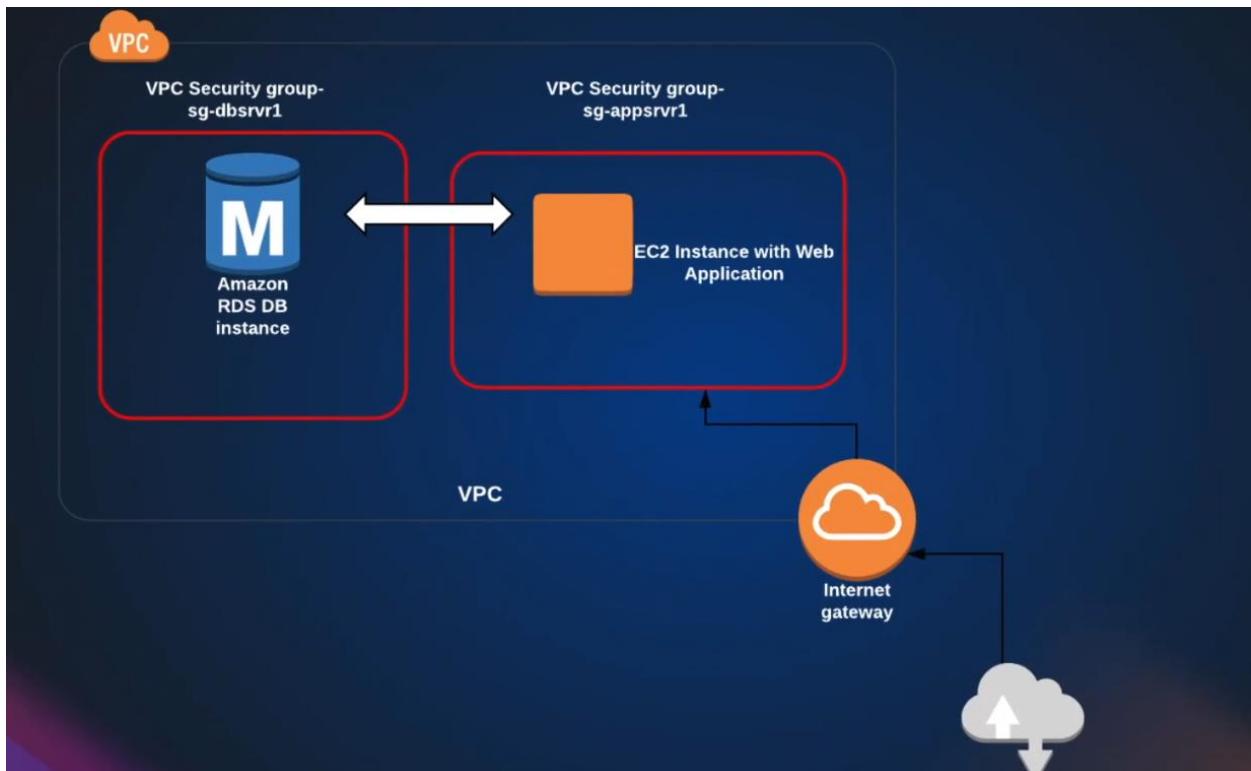
Key Features of AWS RDS:

- **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.

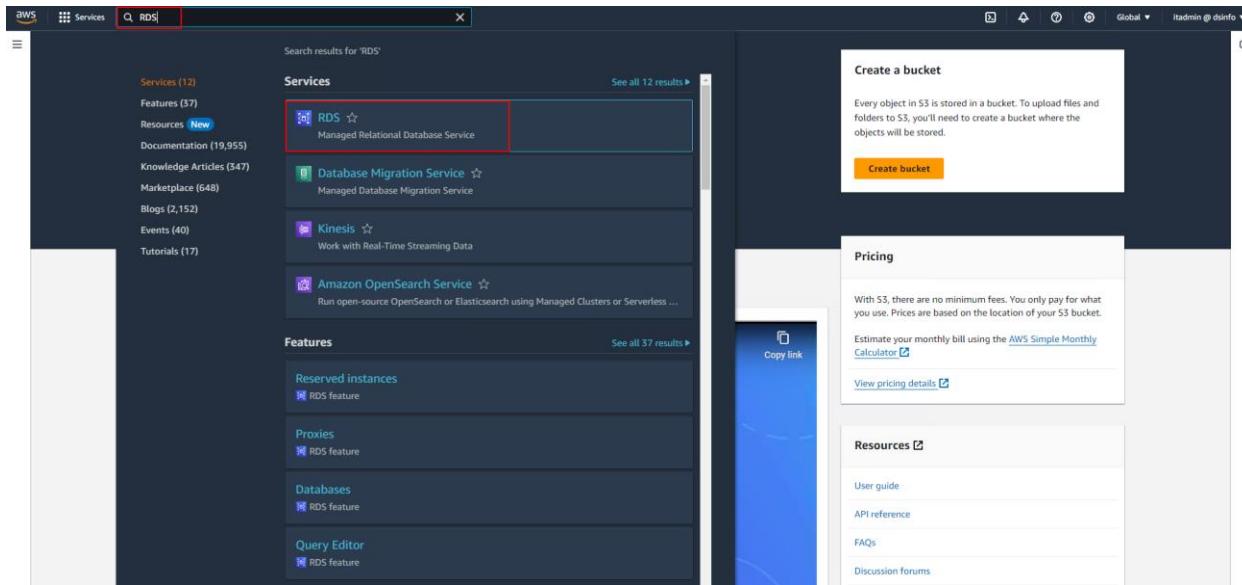
With AWS RDS, you don't need to worry about the complexity of managing database software or infrastructure — AWS takes care of it for you.



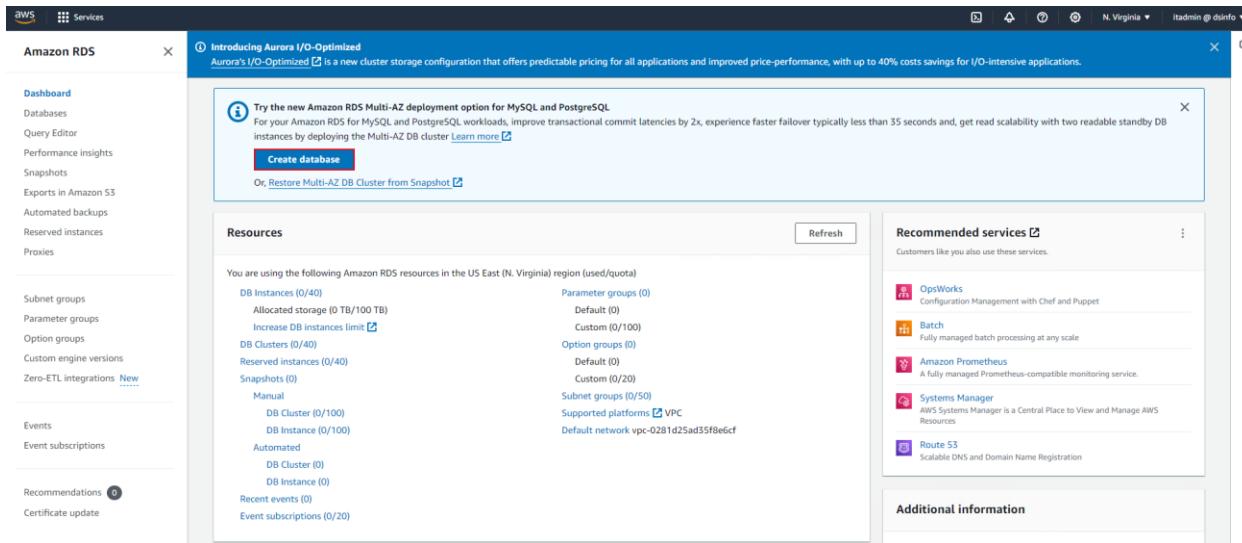
How to Set Up AWS RDS

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

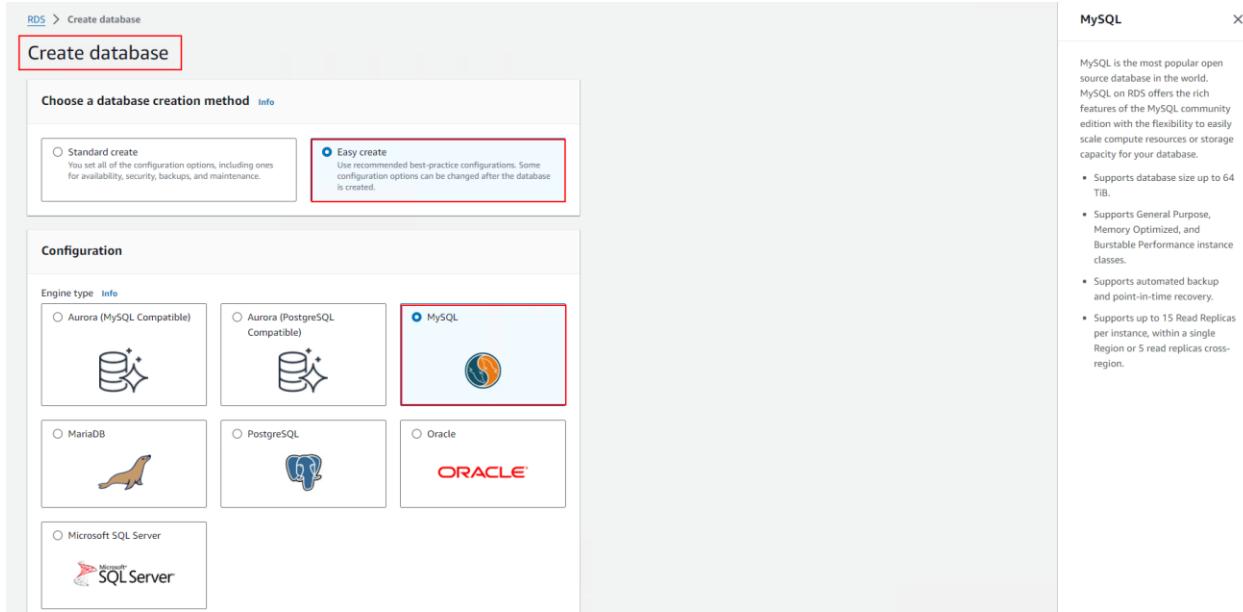
To use **RDS**, type **RDS** in the search bar and then click on **RDS** from the results.



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



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



RDS > Create database

Create database

Choose a database creation method [Info](#)

Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.

Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Configuration

Engine type [Info](#)

Aurora (MySQL Compatible) 

Aurora (PostgreSQL Compatible) 

MySQL 

MariaDB 

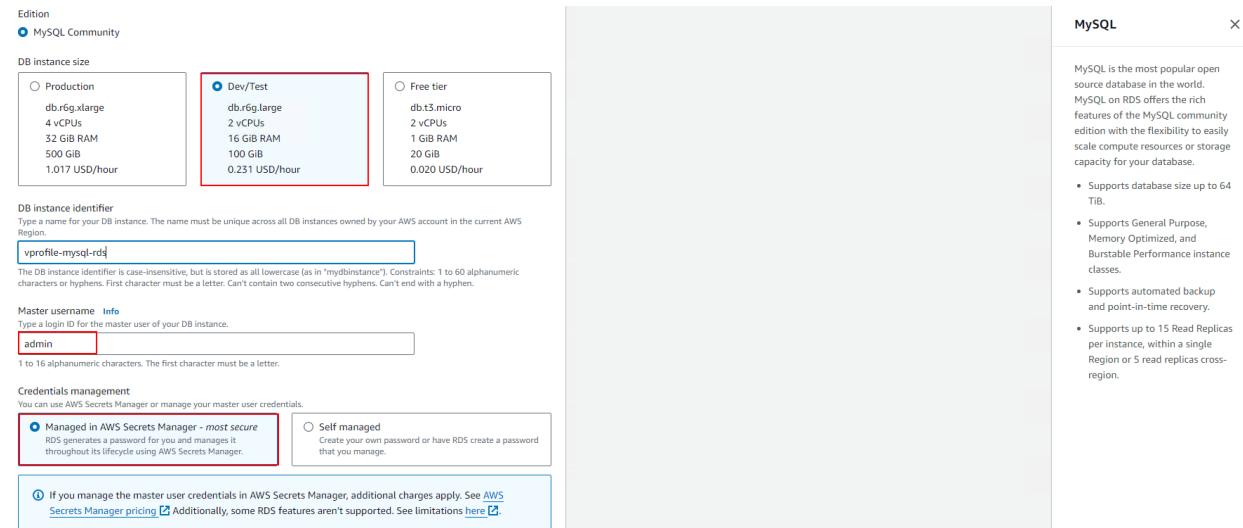
PostgreSQL 

Oracle 

Microsoft SQL Server 

MySQL

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



Edition [Info](#)

MySQL Community

DB instance size

Production db.r6g.xlarge
4 vCPUs
32 GB RAM
500 GiB
0.017 USD/hour

Dev/Test db.r6g.large
2 vCPUs
16 GiB RAM
100 GiB
0.021 USD/hour

Free tier db.t3.micro
2 vCPUs
1 GiB RAM
20 GiB
0.020 USD/hour

DB instance identifier [Info](#)
Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

Master username [Info](#)
Type a login ID for the master user of your DB instance.

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management
You can use AWS Secrets Manager or manage your master user credentials.

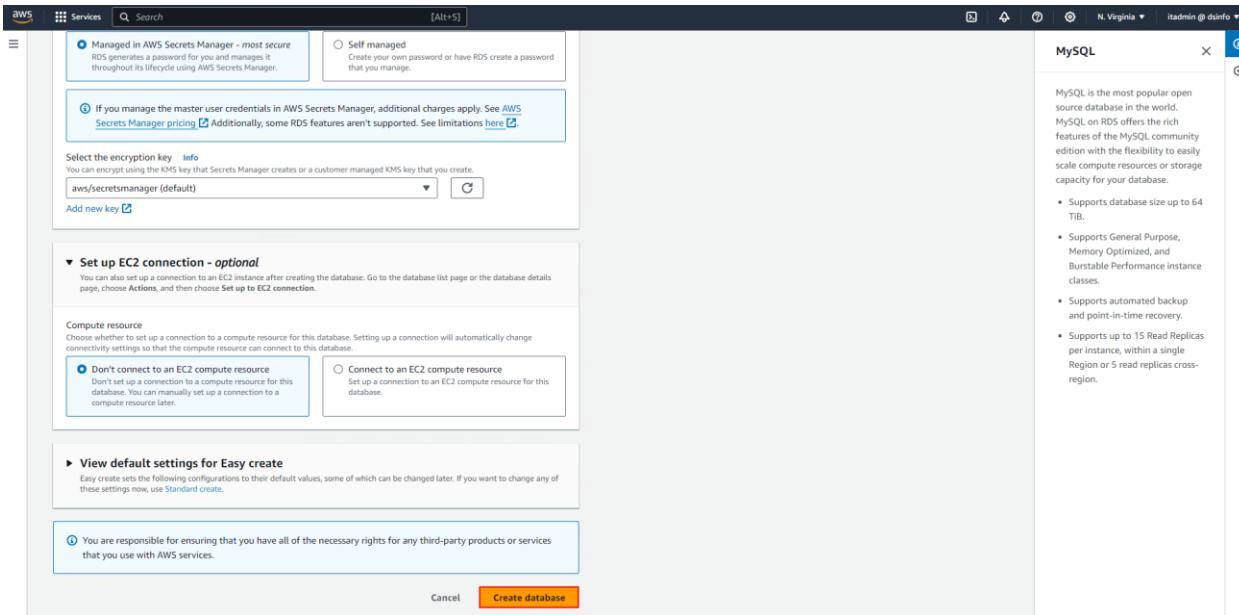
Managed in AWS Secrets Manager - most secure
RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

Self managed
Create your own password or have RDS create a password that you manage.

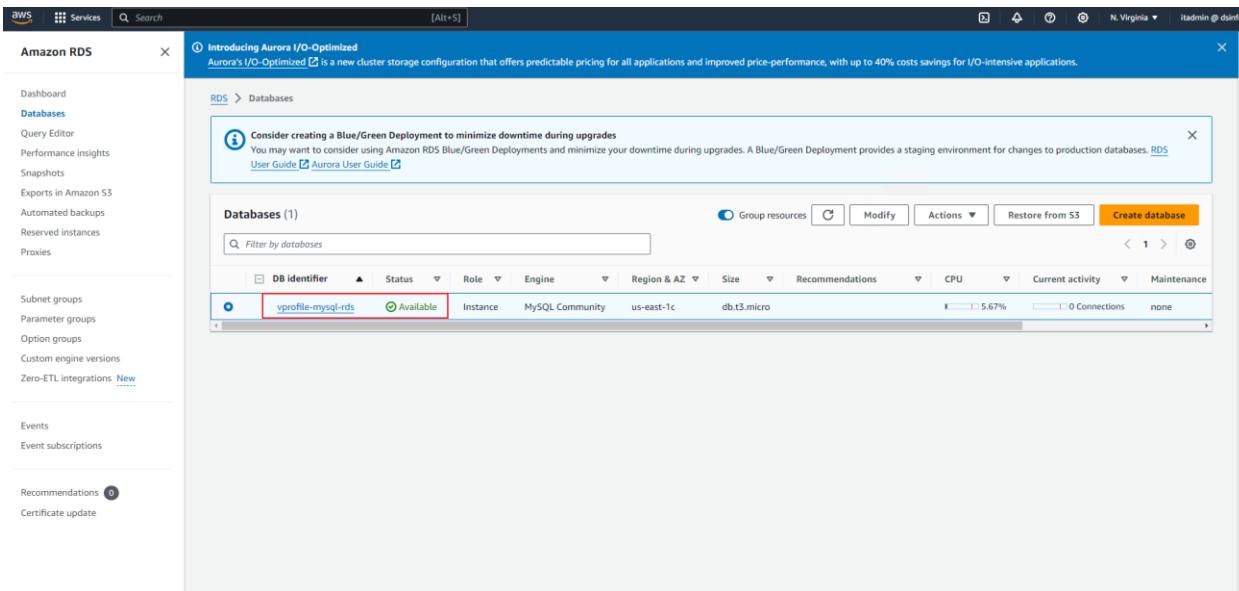
If you manage the master user credentials in AWS Secrets Manager, additional charges apply. See [AWS Secrets Manager pricing](#). Additionally, some RDS features aren't supported. See limitations [here](#).

MySQL

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**.

The screenshot shows two configuration sections for an AWS RDS instance.

Instance configuration: This section allows selecting the DB instance class. A red box highlights the "Burstable classes (includes t classes)" option, which is selected. Below it, a dropdown menu shows "db.t3.micro" with details: 2 vCPUs, 1 GiB RAM, and Network: 2,085 Mbps.

Storage: This section allows defining storage type and size. A red box highlights the "General Purpose SSD (gp2)" storage type and the "Allocated storage" input field set to 20 GiB. A note below states: "The minimum value is 20 GiB and the maximum value is 6,144 GiB".

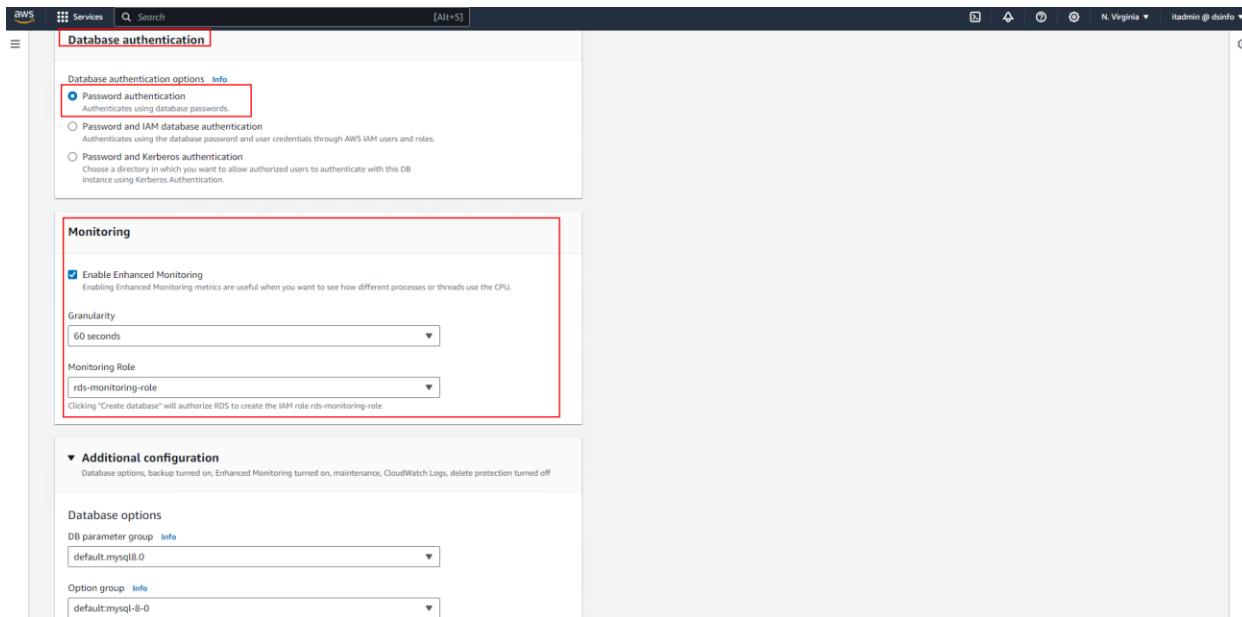
In this section, you can define the **maximum size** of the database and specify whether you want your database to be **replicated across multiple Availability Zones** or not.

The screenshot shows two configuration sections for an AWS RDS instance.

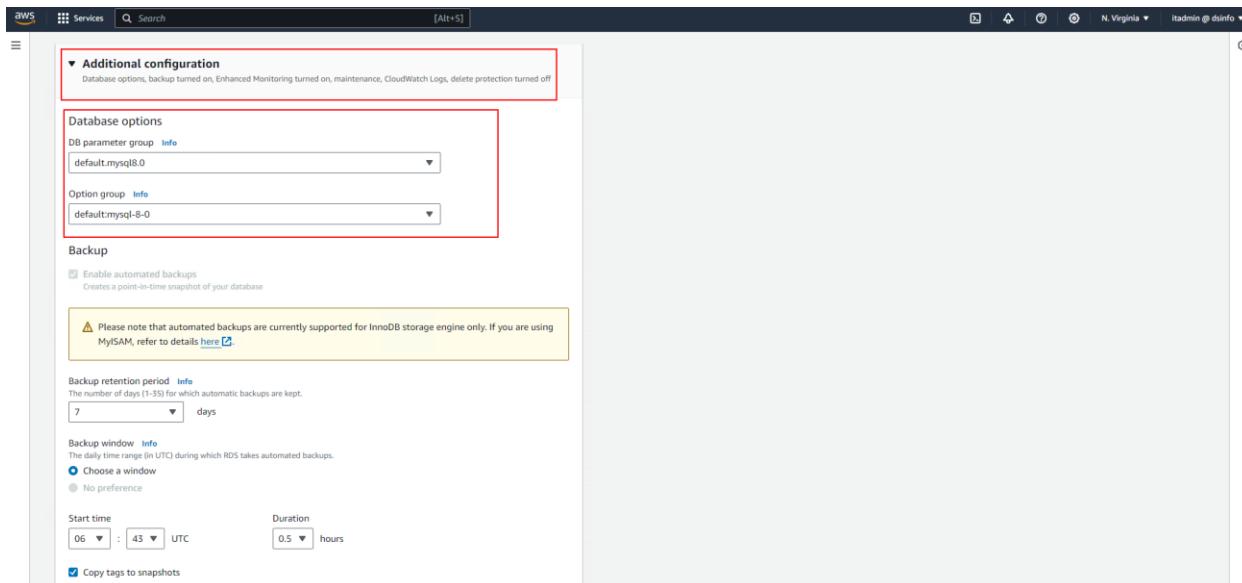
Storage autoscaling: This section enables storage scaling. A red box highlights the "Enable storage autoscaling" checkbox, which is checked. Below it, a dropdown menu shows "1000 GiB" as the maximum storage threshold.

Availability & durability: This section specifies deployment options. A red box highlights the "Do not create a standby instance" option, which is selected. Other options shown are "Create a standby instance (recommended for production usage)" and "Multi-AZ deployment".

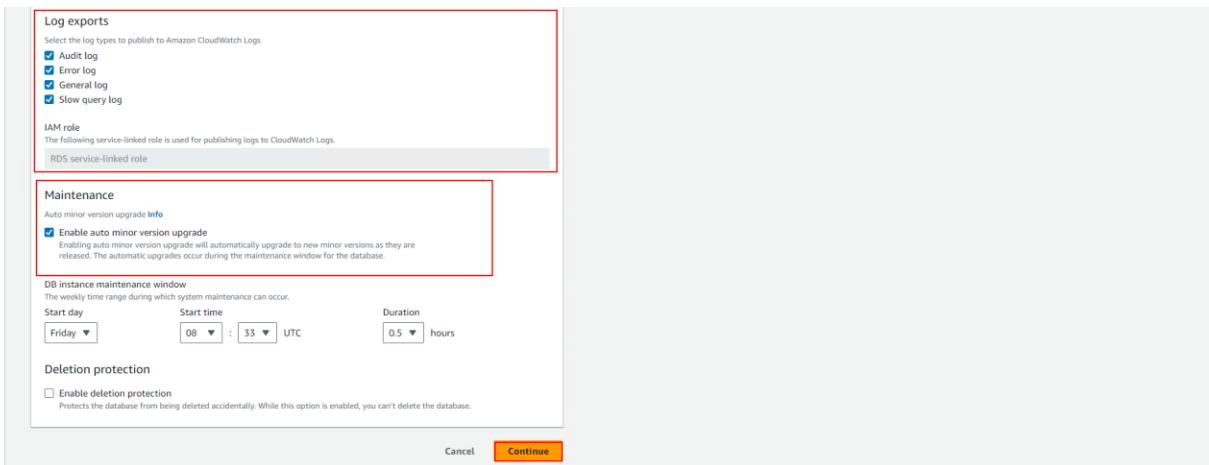
In this section, you can configure **monitoring** for the database.



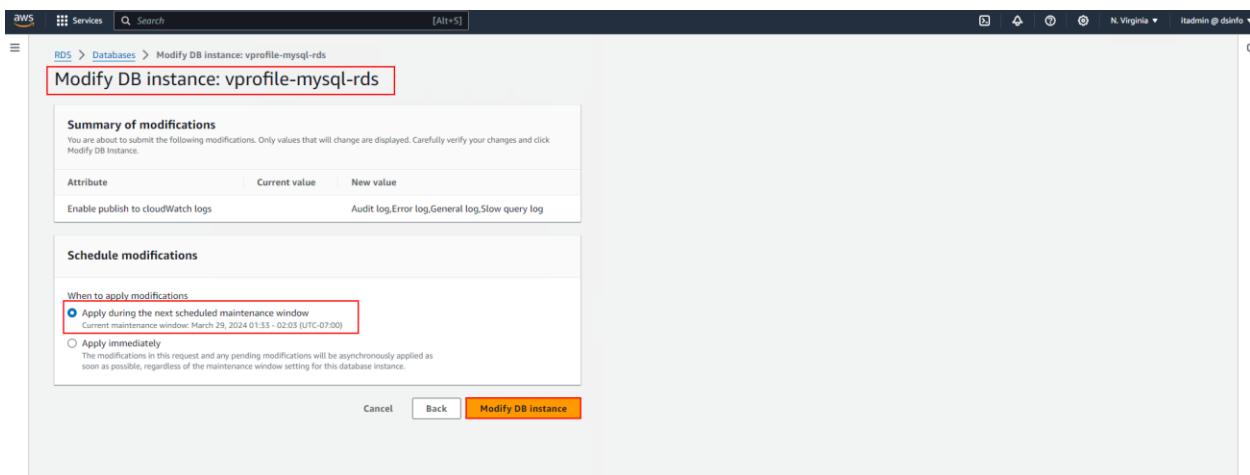
In this section, you can specify the **version** of your database.



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.

