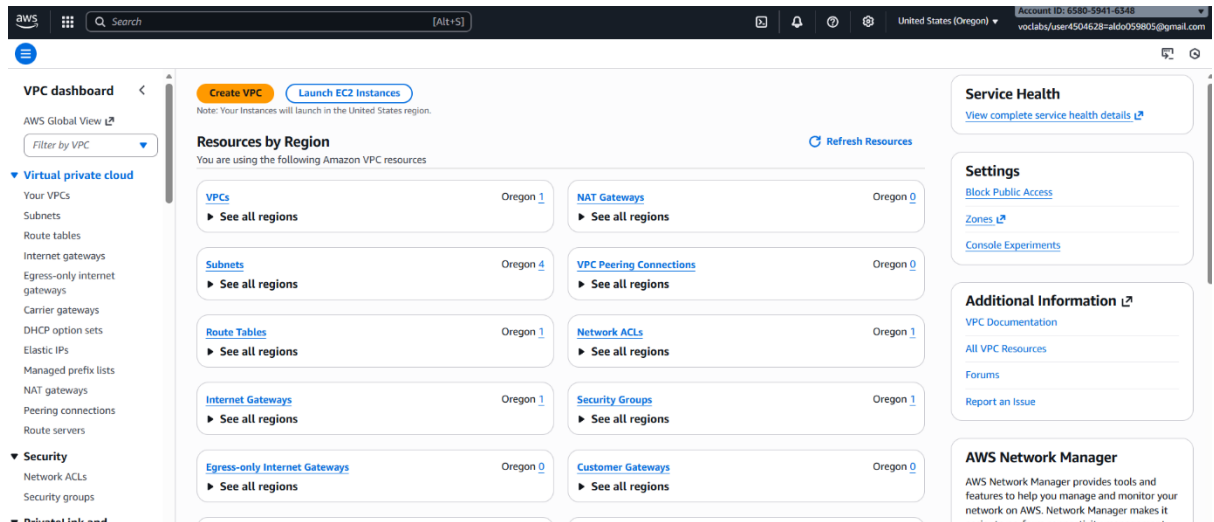


# Simple Web Deployment on AWS EC2 Project

Step-by-step guide deploy an index.html page using nginx

## Step 1 : Create VPC

1. Head over to the VPC console
2. Click Create VPC



3. In VPC setting
  1. Chose VPC only
  2. In Name tag give your VPC unique name like “aldo-vpc”
  3. In IPv4 CIDR Block , Chose IPv4 Manual Input
  4. And in Column IPv4 CIDR input 10.0.0.0/16
  5. In IPv6 CIDR Block , Chose No IPv6 CIDR Block

≡ VPC > Your VPCs > Create VPC

### Create VPC [Info](#)

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

#### VPC settings

##### Resources to create [Info](#)

Create only the VPC resource or the VPC and other networking resources.

☒ VPC only ☐ VPC and more

##### Name tag - optional

Creates a tag with a key of 'Name' and a value that you specify.

aldo-vpc

##### IPv4 CIDR block [Info](#)

☒ IPv4 CIDR manual input  
☐ IPAM-allocated IPv4 CIDR block

##### IPv4 CIDR

10.0.0.0/16

CIDR block size must be between /16 and /28.

##### IPv6 CIDR block [Info](#)

☒ No IPv6 CIDR block  
☐ IPAM-allocated IPv6 CIDR block  
☐ Amazon-provided IPv6 CIDR block  
☐ IPv6 CIDR owned by me

4. Then Scroll to the bottom and leave all default
5. The Click Create VPC
6. And you Succesfully creat a VPC

**Tenancy** [Info](#)

Default

**VPC encryption control (\$)** - new [Info](#)

Monitor mode provides visibility into encryption status without blocking traffic. Enforce mode prevents unencrypted traffic. Additional charges apply [L](#)

☒ **None**

☐ **Monitor mode**  
See which resources in your VPC are unencrypted but allow the creation of unencrypted resources.

☐ **Enforce mode**  
Requires all resources, except exclusions, in your VPC to be encryption-capable and blocks creation of unencrypted resources.

**Tags**  
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.

**Key**

Q Name

**Value - optional**

Q aldo-vpc

[Remove tag](#)

[Add tag](#)

You can add 49 more tags

[Cancel](#) [Preview code](#) [Create VPC](#)

**VPC dashboard** <

AWS Global View [L](#)

Filter by VPC

**Virtual private cloud**

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

NAT gateways

Peering connections

Route servers

**Security**

Network ACLs

Security groups

**vpc-Odc49f44fca193681 / aldo-vpc** [Actions](#)

**Details** [Info](#)

**VPC ID** vpc-Odc49f44fca193681

**State** [Available](#)

**DNS resolution** Enabled

**Main network ACL** acl-0b9af0870a71faa13

**IPv6 CIDR (Network border group)** -

**Encryption control ID** -

**Tenancy** default

**Default VPC** No

**Network Address Usage metrics** Disabled

**Encryption control mode** -

**Block Public Access** [Off](#)

**DHCP option set** dopt-051838db60483e061

**IPv4 CIDR** 10.0.0.0/16

**Route 53 Resolver DNS Firewall rule groups** -

**DNS hostnames** Disabled

**Main route table** rtb-0efc140f48774f67c

**IPv6 pool** -

**Owner ID** [658059416348](#)

**Resource map** [Info](#)

**VPC** Your AWS virtual network

**Subnets (0)** Subnets within this VPC

**Route tables (1)** Route network traffic to resources

**Network Connections** Connections to other networks

## Step 2 : Create a Subnet

1. In VPC Dashboard you Click Subnets
2. And then Click Create Subnet

**VPC dashboard** <

AWS Global View [L](#)

Filter by VPC

**Virtual private cloud**

Your VPCs

**Subnets**

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

NAT gateways

Peering connections

Route servers

**Security**

Network ACLs

Security groups

**Subnets (4)** [Info](#)

Last updated 2 minutes ago [Actions](#) [Create subnet](#)

Find subnets by attribute or tag

| Name | Subnet ID                | State                     | VPC                   | Block Public...     | IPv4 CIDR      | IPv6 CIDR |
|------|--------------------------|---------------------------|-----------------------|---------------------|----------------|-----------|
| -    | subnet-0f8bb3bf8b8de3fa  | <a href="#">Available</a> | vpc-0bf71d4a3c5a6bd1c | <a href="#">Off</a> | 172.31.0.0/20  | -         |
| -    | subnet-0f3c84d79905d896c | <a href="#">Available</a> | vpc-0bf71d4a3c5a6bd1c | <a href="#">Off</a> | 172.31.16.0/20 | -         |
| -    | subnet-071c26238f16631d0 | <a href="#">Available</a> | vpc-0bf71d4a3c5a6bd1c | <a href="#">Off</a> | 172.31.32.0/20 | -         |
| -    | subnet-0fbdcd8899cae51bc | <a href="#">Available</a> | vpc-0bf71d4a3c5a6bd1c | <a href="#">Off</a> | 172.31.48.0/20 | -         |

**Select a subnet**

3. On the page create VPC , chose a coloumn VPC ID and chose you VPC name the one that has been made earlier

Create subnet [Info](#)

**VPC**  
VPC ID  
Create subnets in this VPC.  
Select a VPC

Q |

vpc-0bf71d4a3c5a6bd1c  
172.31.0.0/16 (default)

vpc-0dc49f44fca193681 (aldo-vpc)  
10.0.0.0/16

Select a VPC first to create new subnets.

Add new subnet

Cancel Create subnet

4. In Subnets Setting
  1. Create You subnet name in here iam using “Public Subnet”
  2. Leave default Availability zone
  3. And In coloumn IPv4 Subnet CIDR Block input 10.0.0.0/24

**Subnet settings**  
Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

**Subnet name**  
Create a tag with a key of 'Name' and a value that you specify.  
public subnet  
The name can be up to 256 characters long.

**Availability Zone** [Info](#)  
Choose the zone in which your subnet will reside, or let Amazon choose one for you.  
No preference

**IPv4 VPC CIDR block** [Info](#)  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.  
10.0.0.0/16

**IPv4 subnet CIDR block**  
10.0.0.0/24 256 IPs

5. Scroll to the bottom and click create subnet
6. Then you already create a subnet

▼ Tags - optional

| Key    | Value - optional |
|--------|------------------|
| Q Name | Q public subnet  |

Add new tag

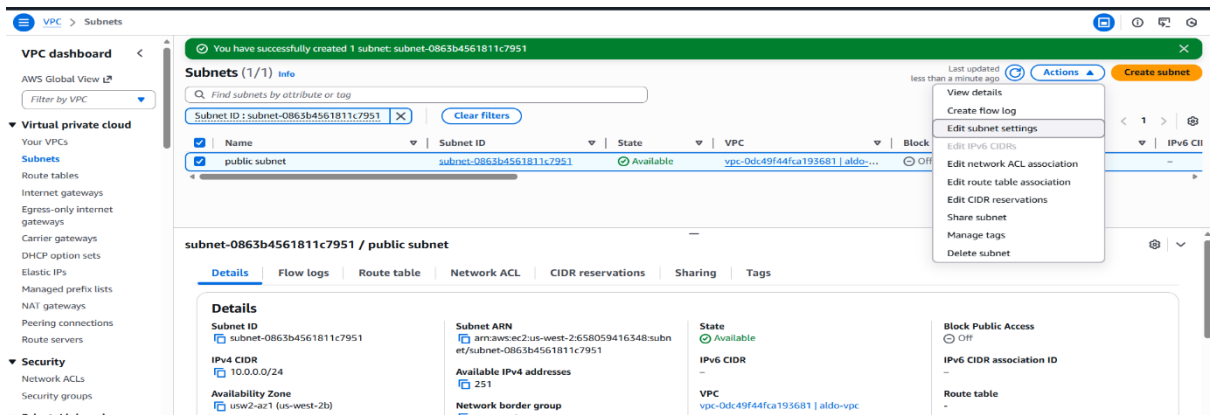
You can add 49 more tags.

Remove

Add new subnet

Cancel Create subnet

7. Then you have to edit subnet setting
8. Click check box in your subnet then click Action
9. And chose Edit subnet settings



10. On Page subnet settings you will see Auto-assign IP setting and Resource-base name (RBN) settings , chose check box enable auto-assign and enable resource name

11. Then click save

**Edit subnet settings** [Info](#)

**Subnet**

Subnet ID  
subnet-0863b4561811c7951

Name  
public subnet

**Auto-assign IP settings** [Info](#)

Enable AWS to automatically assign a public IPv4 or IPv6 address to a new primary network interface for an instance in this subnet.

☒ Enable auto-assign public IPv4 address [Info](#)

☐ Enable auto-assign customer-owned IPv4 address [Info](#)  
Option disabled because no customer owned pools found.

**Resource-based name (RBN) settings** [Info](#)

Specify the hostname type for EC2 instances in this subnet and optional RBN DNS query settings.

☒ Enable resource name DNS A record on launch [Info](#)

☐ Enable resource name DNS AAAA record on launch [Info](#)

**Hostname type** [Info](#)

☐ Resource name

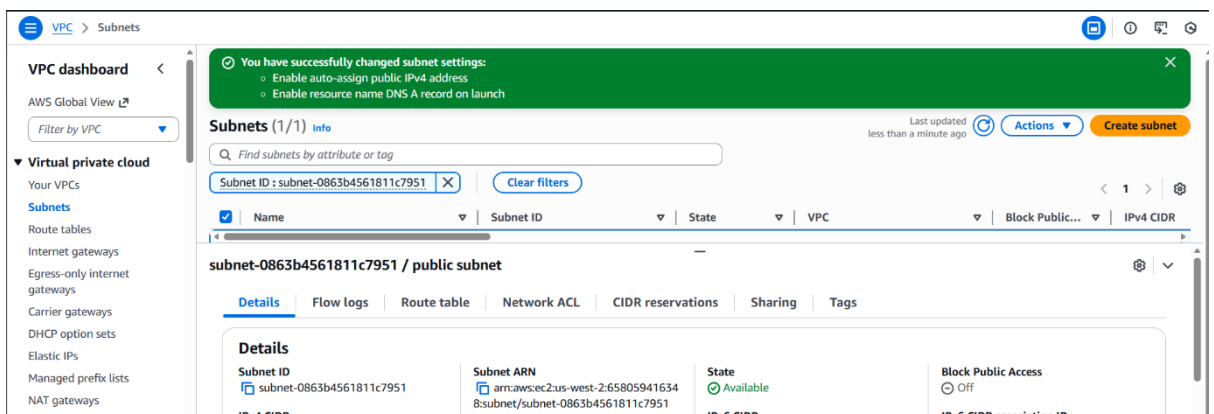
☒ IP name

**DNS64 settings**

Enable DNS64 to allow IPv6-only services in Amazon VPC to communicate with IPv4-only services and networks.

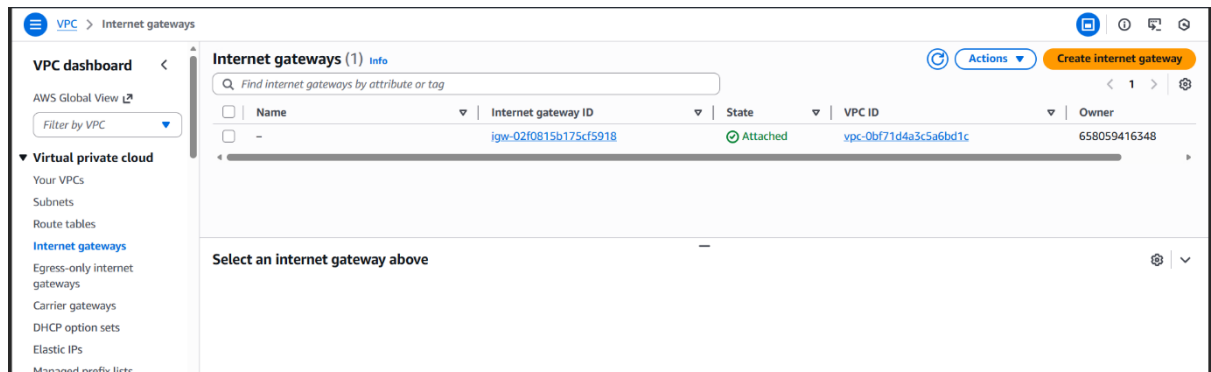
☐ Enable DNS64 [Info](#)

[Cancel](#) [Save](#)

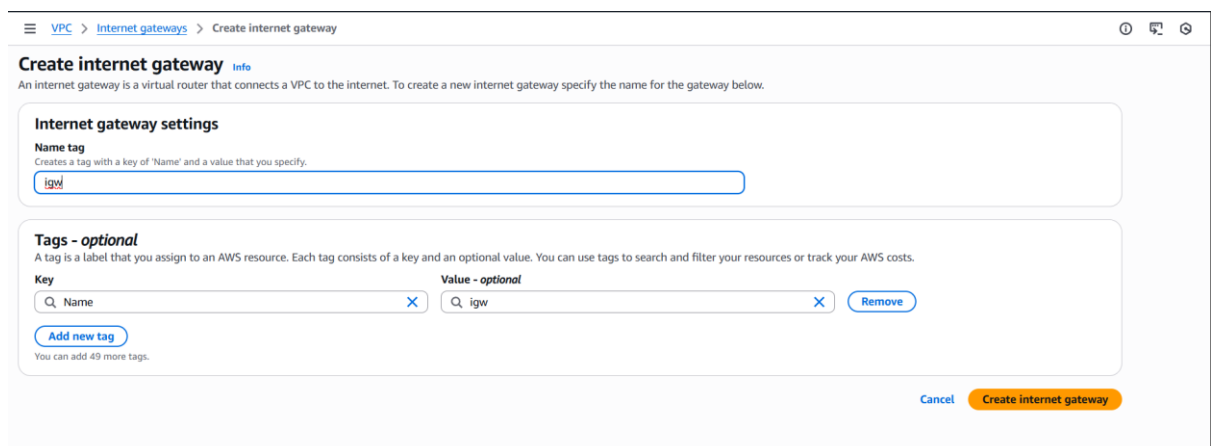


## Step 3 : Create Internet Gateway

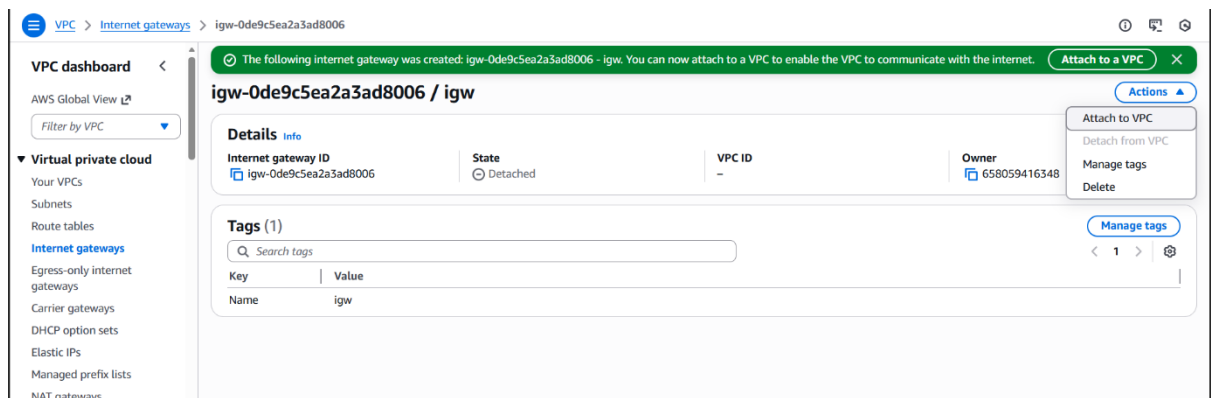
1. In VPC Dashboard you Click Internet gateways
2. Click create internet gateways



3. On Page create internet gateways
4. In column name tag give your name internet gateways , in here i using "igw"
5. Leave all default
6. Click create internet gateways

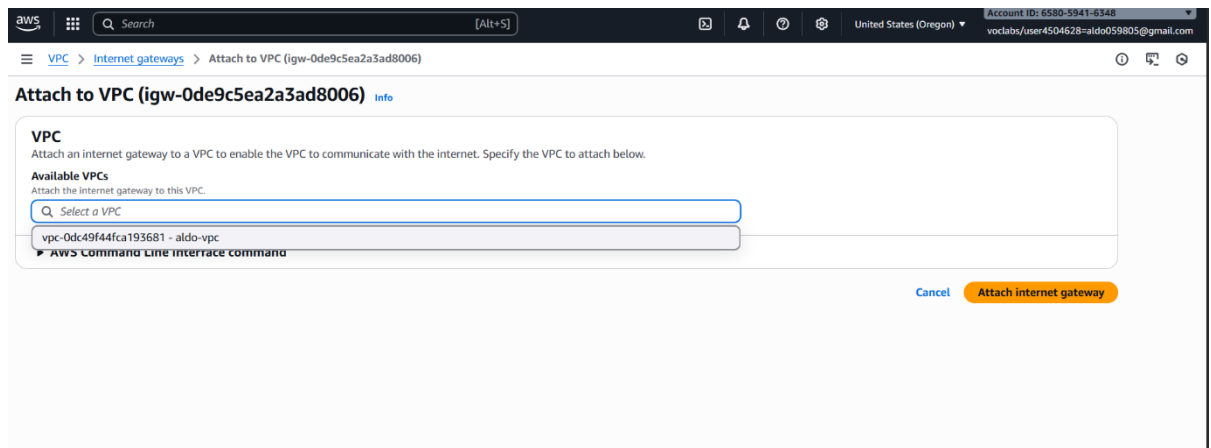


7. After create internet gateway , we have attach internet gateways to VPC
8. On page internet gateways you click action and chose attach to VPC



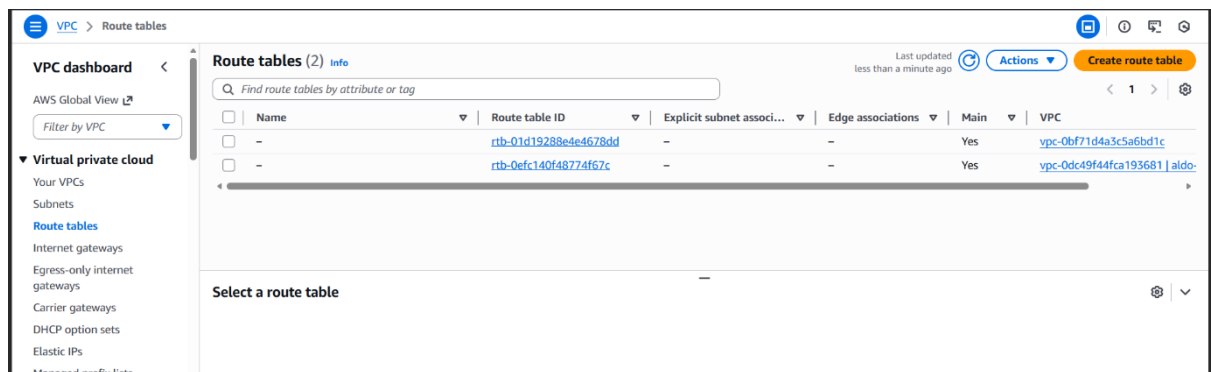
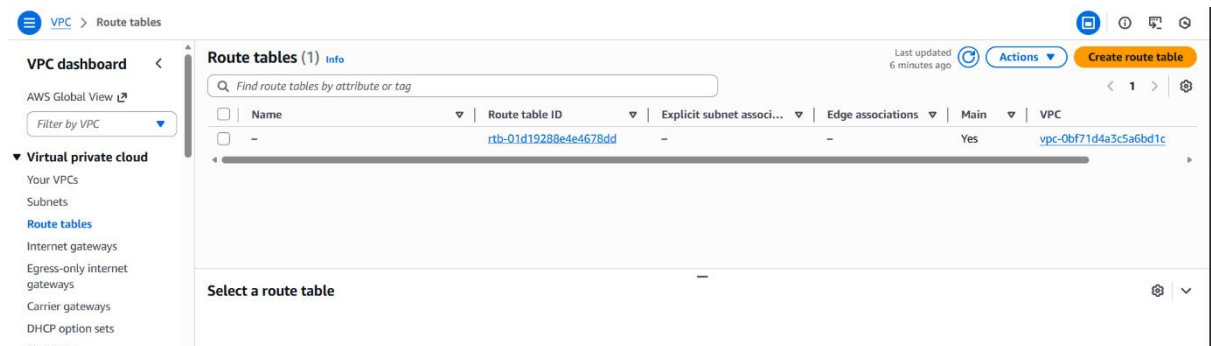
9. On page attach to vpc in available VPCs chose your VPC

## 10. An then click attach internet gateways



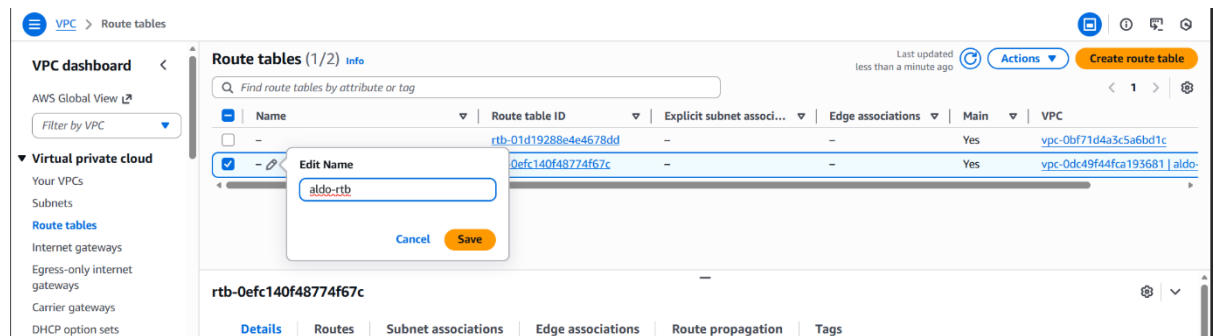
## Step 4 : Create Route Tables

1. In VPC dashboard click Route tables
2. You will see there only have 1 route tables
3. Then you click refresh button next to action button
4. Then you will see there now have 2 route tables



5. In route tables column you can see at VPC column there is a vpc name you made before
6. In name column you can give a name to your route table

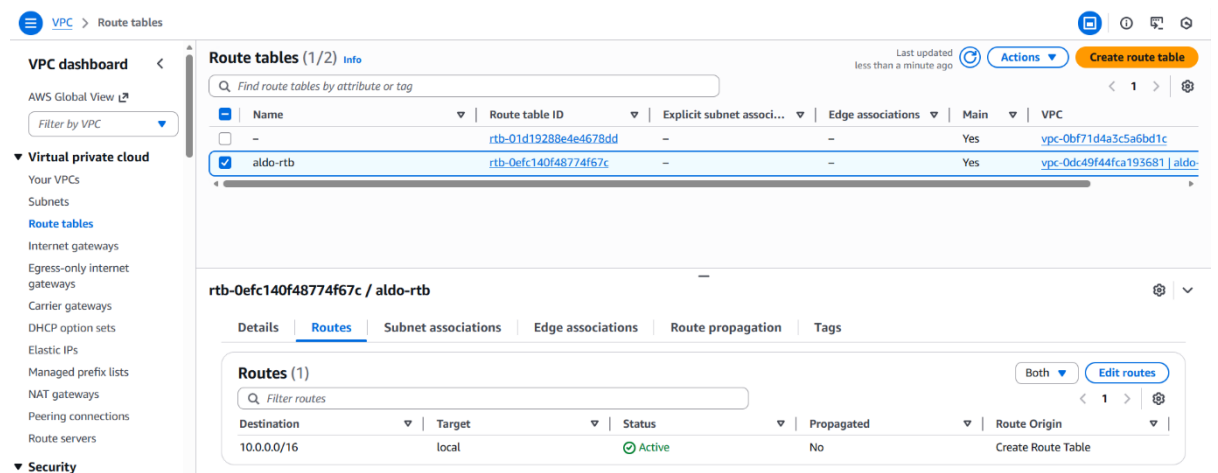
7. Click name column and give your route table name , example “aldo-rtb” and then save



8. After that you have setting routes and subnet association

9. You can see in the page detail page there is a routes page

10. Click route page and then click edit routes

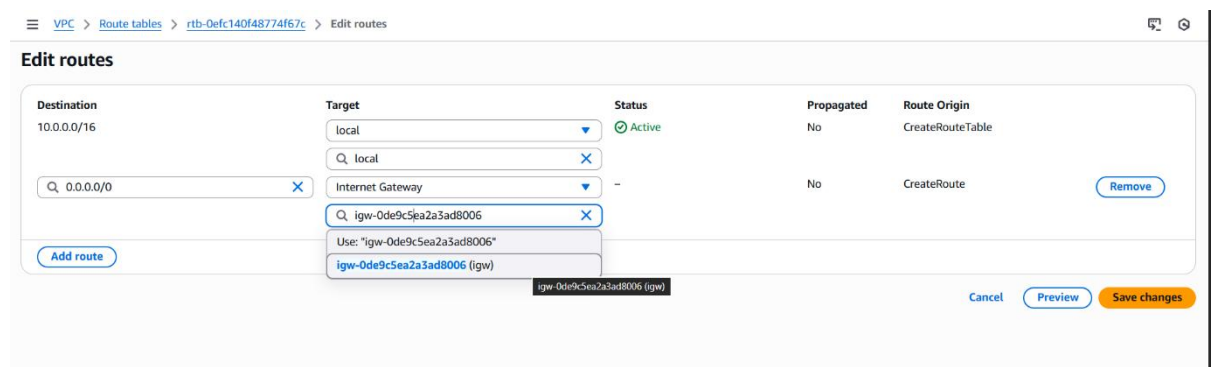


11. On page edit routes you click add route

12. In destination chose 0.0.0.0/0

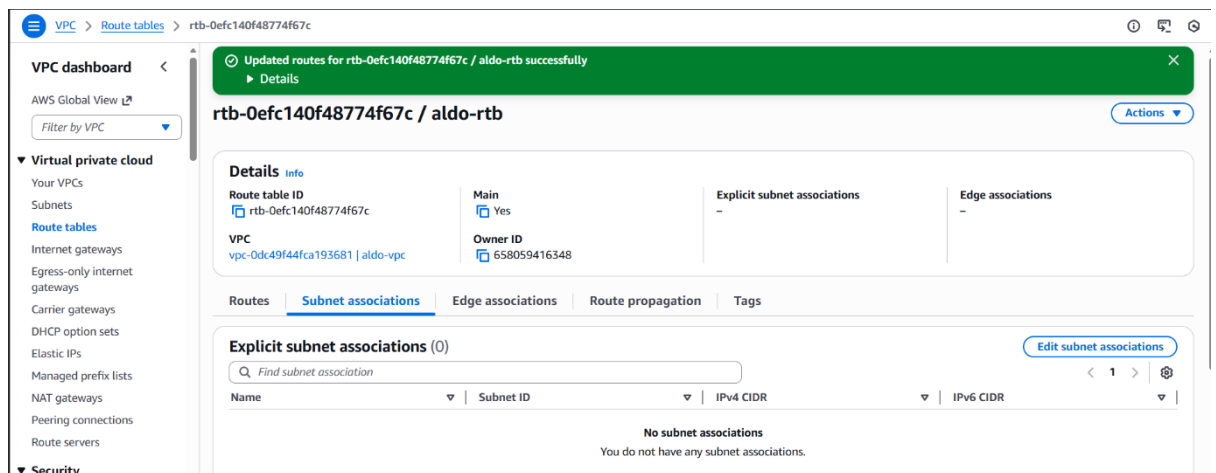
13. And target column chose internet gateways and then chose your internet gateway which has been made before

14. And click save changes

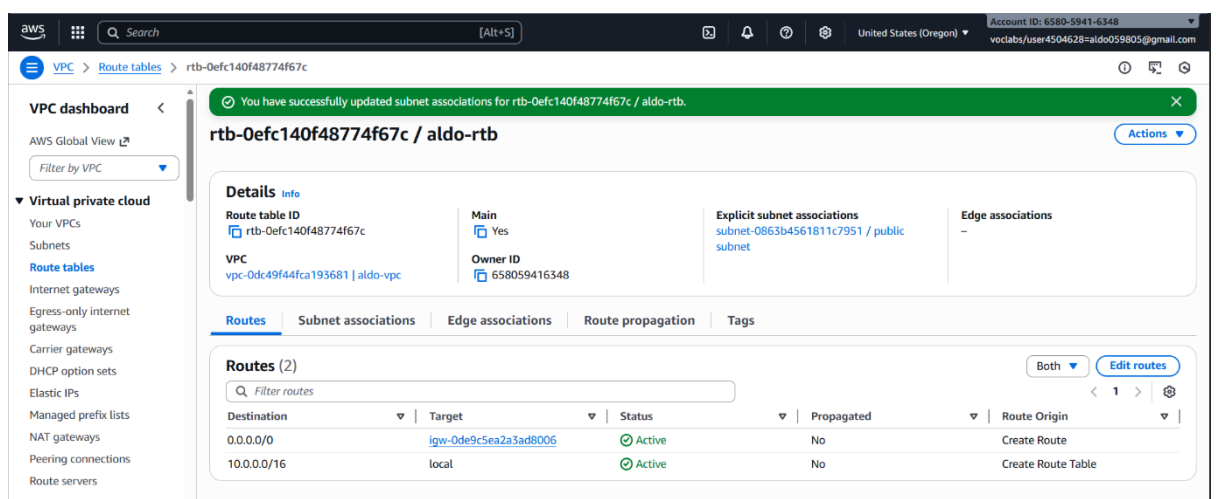
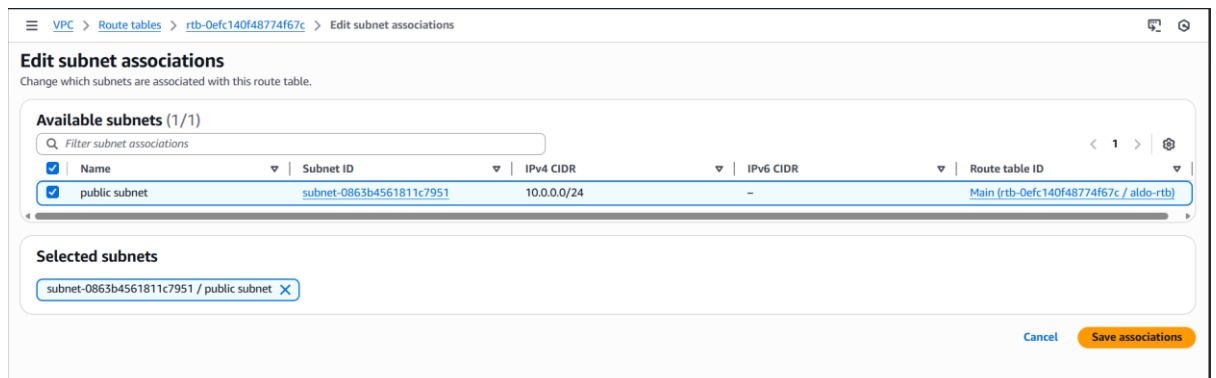


15. Then we go to page Subnet associations

16. Click edit subnet associations

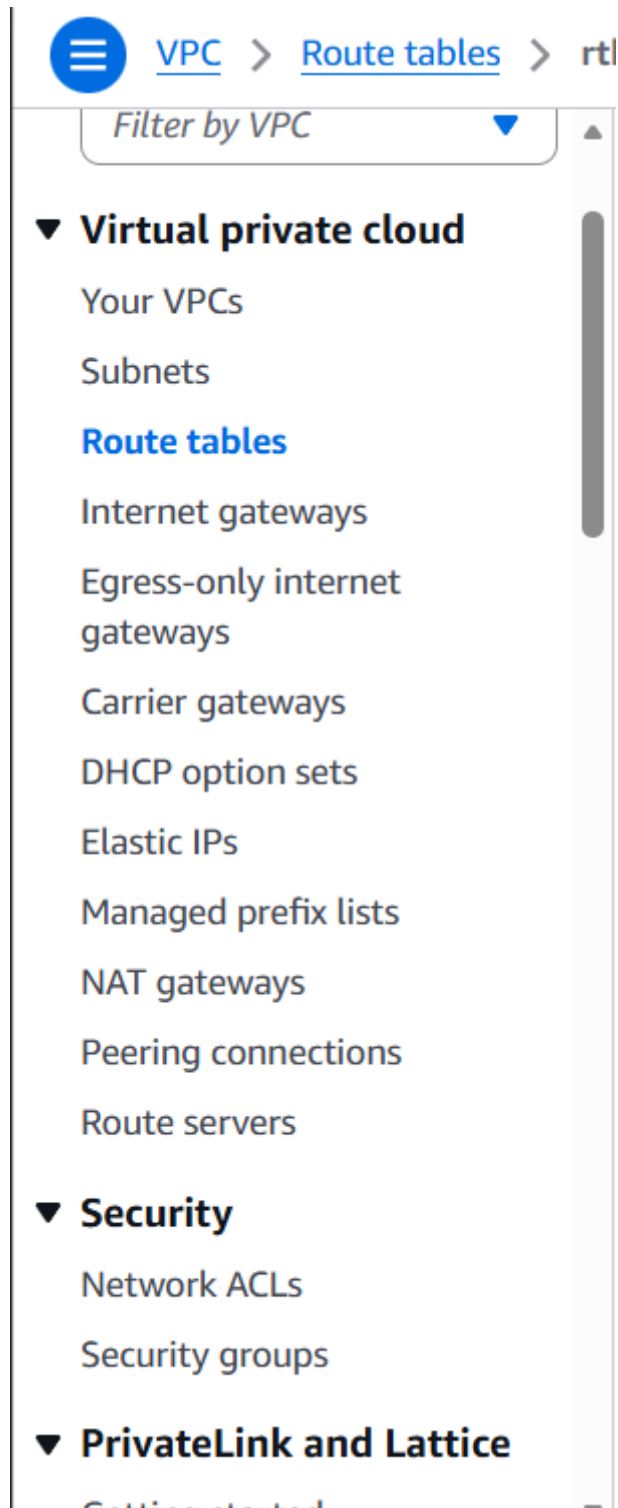


17. On page edit subnet associations you will see you subnet which has been made before
18. Click check box
19. And click save associations

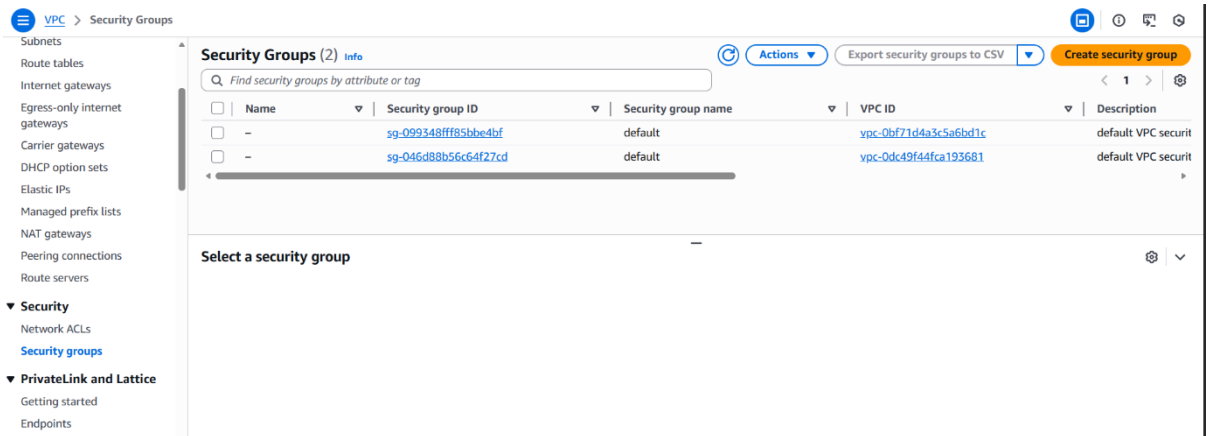


## Step 5 : Create Security Group

1. In AWS console you scroll down until you find Security , and then click security groups



2. On page security group , click create security group



3. On page create security group
4. In basic details
5. Create you security group name , example “aldoSG”
6. Description you can empty this or you can write Allow SSH
7. In vpc click and then chose you vpc name

8. In inbound rules click add rules
9. In type chose http and source chose anywhere IPv4
10. In type chose SSH and source chose anywhere IPv4
11. In type chose https and source chose anywhere IPv4

12. And leave outbound rules default and then click create security group

**Outbound rules** [Info](#)

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

All traffic [▼](#) All [All](#) Custom [▼](#)  [Delete](#)

[Add rule](#)

**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](#)

VPC > Security Groups > sg-0f64eca6b35553612 - aldoSG

Subnets  
Route tables  
Internet gateways  
Egress-only internet gateways  
Carrier gateways  
DHCP option sets  
Elastic IPs  
Managed prefix lists  
NAT gateways  
Peering connections  
Route servers

▼ Security  
Network ACLs  
Security groups

▼ PrivateLink and Lattice  
Getting started  
Endpoints  
Endpoint services  
Service networks

Security group (sg-0f64eca6b35553612 | aldoSG) was created successfully  
Details

**sg-0f64eca6b35553612 - aldoSG** [Actions](#)

**Details**

|                               |   |  |                                 |
|-------------------------------|---|--|---------------------------------|
| Security group name<br>aldoSG | Security group ID<br>sg-0f64eca6b35553612   | Description<br>Allow ssh                   | VPC ID<br>vpc-0dc49f44fca193681 |
| Owner<br>658059416348         | Inbound rules count<br>3 Permission entries | Outbound rules count<br>1 Permission entry |                                 |

[Inbound rules](#) [Outbound rules](#) [Sharing - new](#) [VPC associations - new](#) [Tags](#)

**Inbound rules (3)** [Manage tags](#) [Edit inbound rules](#)

| <input type="checkbox"/> | Name | Security group rule ID | IP version | Type  | Protocol | Port range |
|--------------------------|------|------------------------|------------|-------|----------|------------|
| <input type="checkbox"/> | -    | sgr-07b59c859d1b2c77a  | IPv4       | HTTPS | TCP      | 443        |
| <input type="checkbox"/> | -    | sgr-05137ec1d190593cf  | IPv4       | HTTP  | TCP      | 80         |
| <input type="checkbox"/> | -    | sqr-052a8466b9d3e882   | IPv4       | SSH   | TCP      | 22         |

## Step 6 : Create EC2 Instance

1. On aws console search for EC2

aws

VPC >

Subnets  
Route tables  
Internet gateways  
Egress-only internet gateways  
Carrier gateways  
DHCP option sets  
Elastic IPs  
Managed prefix lists  
NAT gateways  
Peering connections  
Route servers

▼ Security  
Network ACLs  
Security groups

▼ PrivateLink and Lattice  
Getting started  
Endpoints  
Endpoint services  
Service networks

**Services** [Show more](#)

- 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

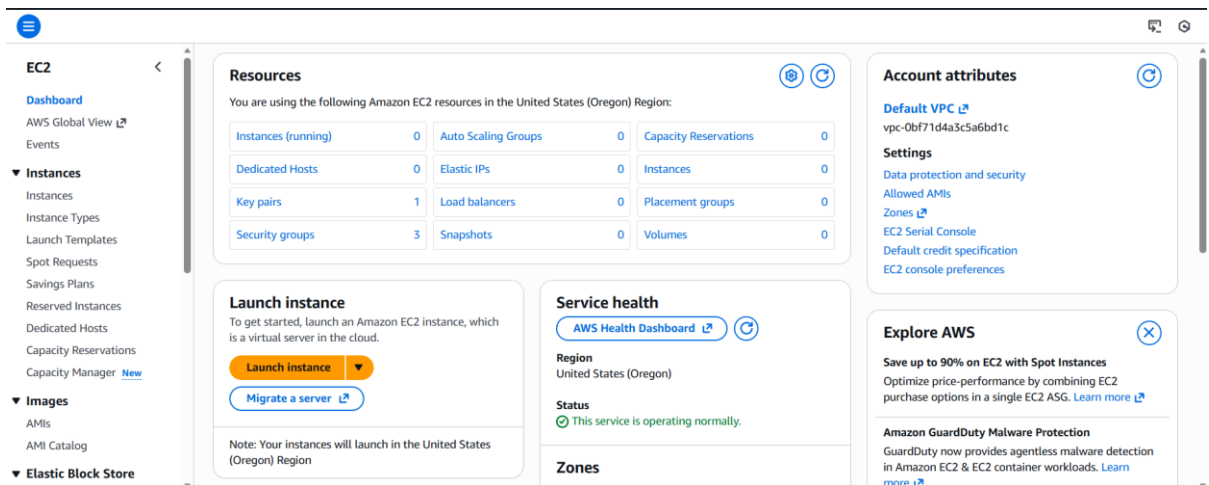
**Features** [Show more](#)

- EC2 Instances** CloudWatch feature
- EC2 Resource Health** CloudWatch feature
- Dashboard**

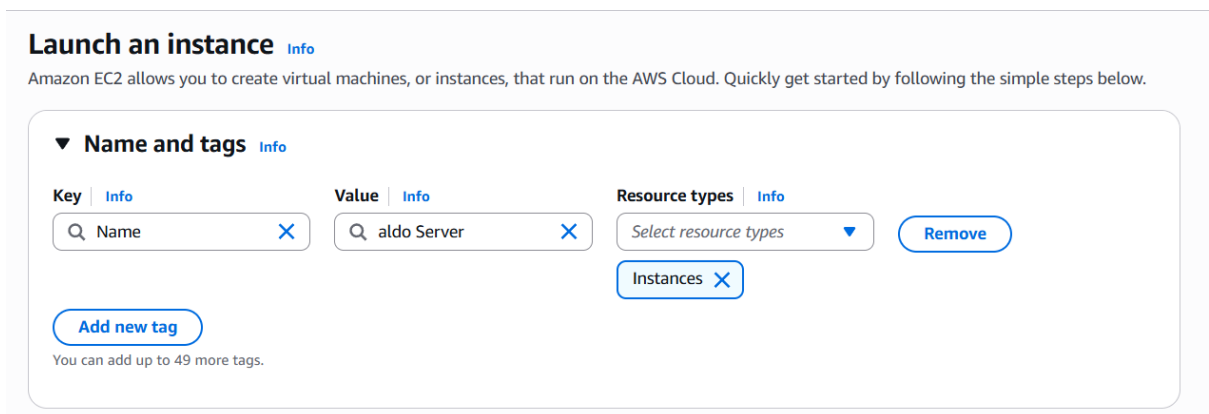
Were these results helpful?

[Yes](#) [No](#)

2. Click EC2
3. On EC2 dashboard page click launch instance



4. On page launch an instance you can create and add your instance name , example “aldo Server”



5. In application and OS Images (Amazon Machine Image) chose Amazon Linux then leave default the rest

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE Linux

SUSE

Debian

debian

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 kernel-6.1 AMI

ami-0ebf411a80b6b22cb (64-bit (x86), uefi-preferred) / ami-0e723566181f273cd (64-bit (Arm), uefi)  
Virtualization: hvm    ENA enabled: true    Root device type: ebs

Free tier eligible

Description

Amazon Linux 2023 (kernel-6.1) 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.

Amazon Linux 2023 AMI 2023.9.20251208.0 x86\_64 HVM kernel-6.1

Architecture

64-bit (x86)

Boot mode

uefi-preferred

AMI ID

ami-0ebf411a80b6b22cb

Publish Date

2025-12-03

Username

ec2-user

Verified provider

- In instance Type chose t3.micro
- And for key pair login chose Proceed without a key pair

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t3.micro

Family: t3    2 vCPU    1 GiB Memory    Current generation: true    On-Demand SUSE base pricing: 0.0104 USD per Hour  
On-Demand Ubuntu Pro base pricing: 0.0139 USD per Hour    On-Demand Windows base pricing: 0.0196 USD per Hour  
On-Demand RHEL base pricing: 0.0392 USD per Hour    On-Demand Linux base pricing: 0.0104 USD per Hour

Free tier eligible

☐ All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required

Proceed without a key pair (Not recommended)

Default value

[Create new key pair](#)

- In network setting , you can click edit

▼ Network settings [Info](#)

Edit

Network [Info](#)

vpc-0bf71d4a3c5a6bd1c

Subnet [Info](#)

No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group
☐ Select existing security group

We'll create a new security group called 'launch-wizard-1' with the following rules:

☒ Allow SSH traffic from

Helps you connect to your instance

Anywhere  
0.0.0.0/0

- In network setting

10. In vpc column you chose your vpc has been made before
11. And subnet will automaticly fill itself or you can chose you subnet has been made before incase you subnet column not automaticly fill itself
12. Auto-assign public IP chose ENABLE
13. Firewall / Security group chose select existing security group , then chose you security group you made before

The screenshot shows the 'Network settings' section in the AWS console. It includes fields for VPC (vpc-0dc49f44fca193681), Subnet (subnet-0863b4561811c7951), and Auto-assign public IP (Enable). The Firewall section has 'Select existing security group' chosen, with 'aldoSG' selected from the common security groups list. A 'Compare security group rules' link is visible. The 'Advanced network configuration' section is partially visible at the bottom.

14. Leave default the rest and click launch instance

The screenshot shows the 'Configure storage' section with '1x 8 GiB gp3' selected. A 'Free tier' notice is displayed. Below, there's a section for 'Advanced details'. On the right, the 'Launch instance' button is highlighted in orange, with a 'Preview code' link below it. A 'Cancel' button is also visible.

15. At page launch successfully just scrool down and then click view all instances

The screenshot shows the 'Launch an instance' page after successful launch. A green banner at the top says 'Success Successfully initiated launch of instance (i-0f5c074f7b1a05411)'. Below, the 'Next Steps' section includes links for 'Create billing and free tier usage alerts', 'Connect to your instance', 'Connect an RDS database', 'Create EBS snapshot policy', 'Manage detailed monitoring', 'Create Load Balancer', 'Create AWS budget', and 'Manage CloudWatch alarms'.

### Manage detailed monitoring

Enable or disable detailed monitoring for the instance. If you enable detailed monitoring, the Amazon EC2 console displays monitoring graphs with a 1-minute period.

[Manage detailed monitoring](#)

### Create Load Balancer

Create an application, network gateway or classic Elastic Load Balancer

[Create Load Balancer](#)

### Create AWS budget

AWS Budgets allows you to create budgets, forecast spend, and take action on your costs and usage from a single location.

[Create AWS budget](#)

### Manage CloudWatch alarms

Create or update Amazon CloudWatch alarms for the instance.

[Manage CloudWatch alarms](#)

View all instances

16. On page instace dashboard you will see you instance , then you see at status check column there is Initializing

The screenshot shows the AWS Management Console for EC2 Instances. The instance 'aldo Server' (ID: i-0f3c074f781a03411) is in the 'Running' state. The 'Status check' column shows 'Initializing'. The 'Alarm status' column has a 'View alarms' link. The 'Availability Zone' is 'us-west-2b'. The 'Public IPv4 DNS' is '34.21'.

17. Just wait 1 – 5 menit then you can click refresh until status check become check passed

18. Then you click check box instance and then click connect , you will redirect to page coonect and then you chose EC2 instance connect and click connect

19. And then you will redirect to amazon linux

The screenshot shows the AWS Management Console for EC2 Instances. The instance 'aldo Server' (ID: i-0f3c074f781a03411) is in the 'Running' state. The 'Status check' column shows '3/3 checks passed'. The 'Alarm status' column has a 'View alarms' link. The 'Availability Zone' is 'us-west-2b'. The 'Public IPv4 DNS' is '34.21'.

The screenshot shows the 'Connect to instance' page in the AWS Management Console. The 'EC2 Instance Connect' tab is selected. The 'Connect using a Public IP' option is chosen. The 'Public IPv4 address' is '34.217.129.204'. The 'Username' is 'ec2-user'. The 'Note' states: 'In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.'

## Step 7 : Install nginx and create a html.index for simple website

1. Run following command

Sudo yum update -y

```
ec2-user@ip-10-0-0-227 ~]$ sudo yum update -y
amazon Linux 2023 Kernel Livepatch repository                               270 kB/s | 29 kB    00:00
Last metadata expiration check: 0:00:01 ago on Tue Dec  9 16:44:45 2025.
Dependencies resolved.
Nothing to do.
Complete!
ec2-user@ip-10-0-0-227 ~]$
```

2. Install nginx

Sudo yum install nginx -y

```
ec2-user@ip-10-0-0-227 ~]$ sudo yum install nginx -y

[ec2-user@ip-10-0-0-227 ~]$ sudo yum install nginx -y
Last metadata expiration check: 0:00:46 ago on Tue Dec  9 16:44:45 2025.
Dependencies resolved.

Package                               Architecture      Version           Repository        Size
Installing:
nginx                                 x86_64            1:1.28.0-1.amzn2023.0.2  amazonlinux      33 k
Installing dependencies:
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch  noarch            18.0.0-12.amzn2023.0.3  amazonlinux      19 k
gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64        x86_64            2.9.1-1.amzn2023.0.3    amazonlinux      308 k
libunwind-1.4.0-5.amzn2023.0.3.x86_64              x86_64            1.4.0-5.amzn2023.0.3    amazonlinux      66 k
nginx-core-1:1.28.0-1.amzn2023.0.2.x86_64          x86_64            1:1.28.0-1.amzn2023.0.2  amazonlinux      686 k
nginx-filesystem-1.28.0-1.amzn2023.0.2.noarch        noarch            1:1.28.0-1.amzn2023.0.2  amazonlinux      9.6 k
nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch         noarch            2.1.49-3.amzn2023.0.3    amazonlinux      21 k

Transaction Summary
Install 7 Packages

Total download size: 1.1 M
Installed size: 3.7 M
Downloading Packages:
(1/7): libunwind-1.4.0-5.amzn2023.0.3.x86_64.rpm                                1.8 MB/s | 66 kB    00:00
(2/7): generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch.rpm                    471 kB/s | 19 kB    00:00
(3/7): gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64.rpm                         6.0 MB/s | 308 kB   00:00
(4/7): nginx-1.28.0-1.amzn2023.0.2.x86_64.rpm                                 1.3 MB/s | 33 kB    00:00
(5/7): nginx-core-1.28.0-1.amzn2023.0.2.x86_64.rpm                             21 MB/s | 686 kB    00:00
(6/7): nginx-filesystem-1.28.0-1.amzn2023.0.2.noarch.rpm                       390 kB/s | 9.6 kB   00:00
(7/7): nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch.rpm                         742 kB/s | 21 kB    00:00
Total                                                                           8.7 MB/s | 1.1 MB   00:00

Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing :
  Running scriptlet: nginx-filesystem-1:1.28.0-1.amzn2023.0.2.noarch              1/1
  Installing : nginx-filesystem-1:1.28.0-1.amzn2023.0.2.noarch                    1/7
  Installing : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch                       2/7
  Installing : libunwind-1.4.0-5.amzn2023.0.3.x86_64                             3/7
  Installing : gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64                       4/7
  Installing : nginx-core-1:1.28.0-1.amzn2023.0.2.x86_64                         5/7
  Installing : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch                  6/7
  Installing : nginx-1:1.28.0-1.amzn2023.0.2.x86_64                             7/7

Installed:
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch  gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64  libunwind-1.4.0-5.amzn2023.0.3.x86_64  nginx-1:1.28.0-1.amzn2023.0.2.x86_64
nginx-core-1:1.28.0-1.amzn2023.0.2.x86_64        nginx-filesystem-1:1.28.0-1.amzn2023.0.2.noarch  nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

Complete!
ec2-user@ip-10-0-0-227 ~]$
```

3. Start and enable nginx

Sudo systemctl start nginx

Sudo systemctl enable nginx

```
[ec2-user@ip-10-0-0-227 ~]$ sudo systemctl start nginx
[ec2-user@ip-10-0-0-227 ~]$ sudo systemctl enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
[ec2-user@ip-10-0-0-227 ~]$
```

#### 4. Create index html

Sudo nano /usr/share/nginx/html/index.html

```
ec2-user@ip-10-0-0-227 ~]$ sudo nano /usr/share/nginx/html/index.html
```

#### 5. Create or copy paste you website code using html

```
GNU nano 2.9.3 /usr/share/nginx/html/index.html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>DevOps Portfolio Project</title>
  <style>
    body {
      margin: 0;
      font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
      background: #0e0f11;
      color: #f2f2f2;
      display: flex;
      flex-direction: column;
      align-items: center;
      justify-content: center;
      height: 100vh;
      text-align: center;
    }

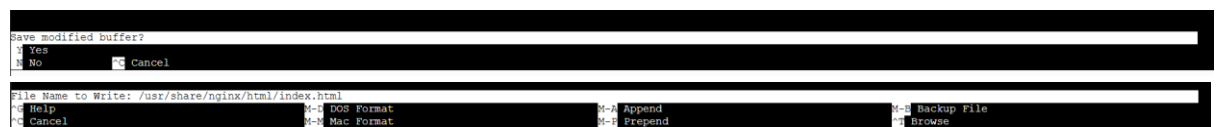
    h1 {
      font-size: 2.8rem;
      margin-bottom: 10px;
      color: #4ea3ff;
    }

    p {
  
```

#### 6. Then press CTRL + X

#### 7. Then press Y

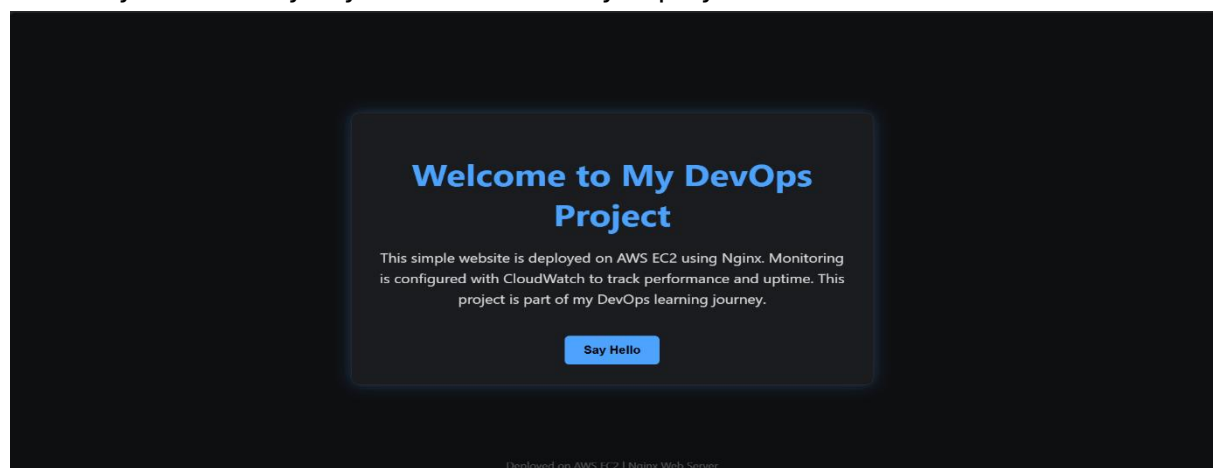
#### 8. Then enter



#### 9. Go back to page AWS console then go to EC2 instance in detail page you will see IP public

#### 10. Click ip public then the new page will be open

#### 11. And you can see you website already deployed



**By completing this project, I learned:**

- How to launch and configure an EC2 instance
- How to manage Linux services (systemctl)
- How to install and configure a web server
- How to deploy static content
- How to configure firewall/security groups
- How to work with Nginx directory structure