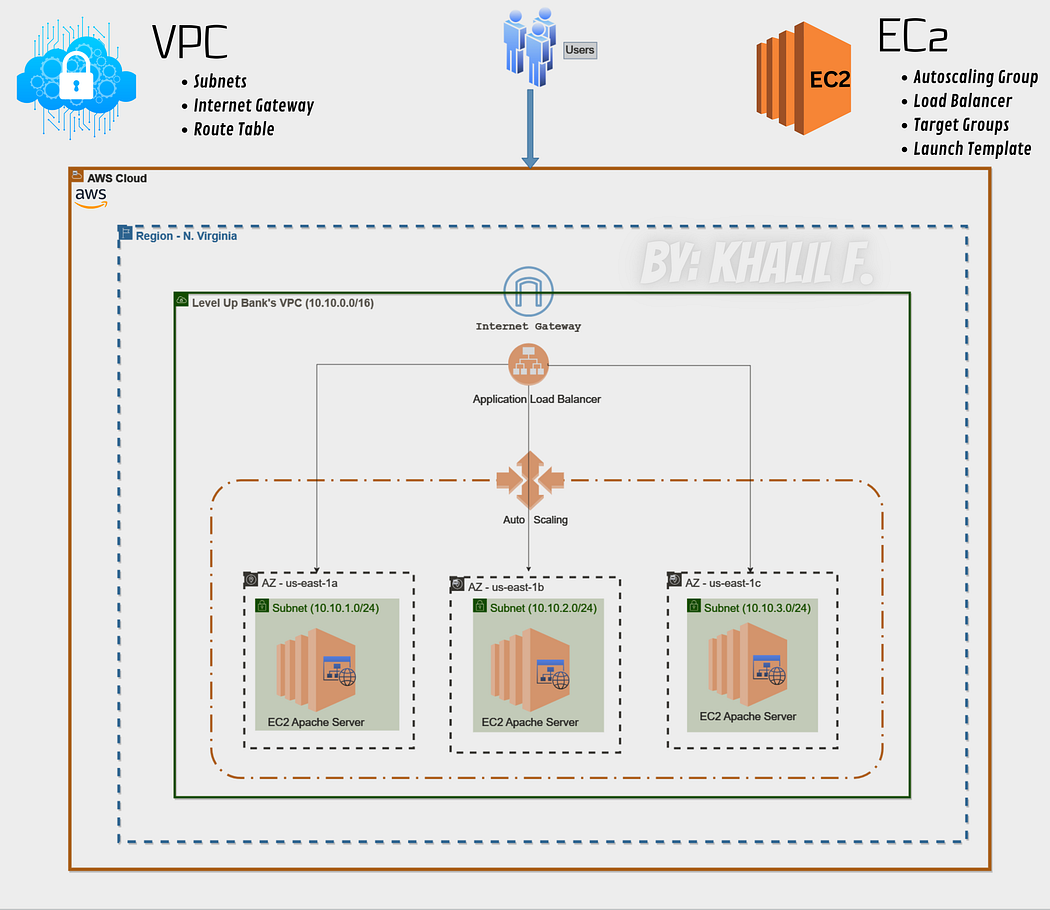
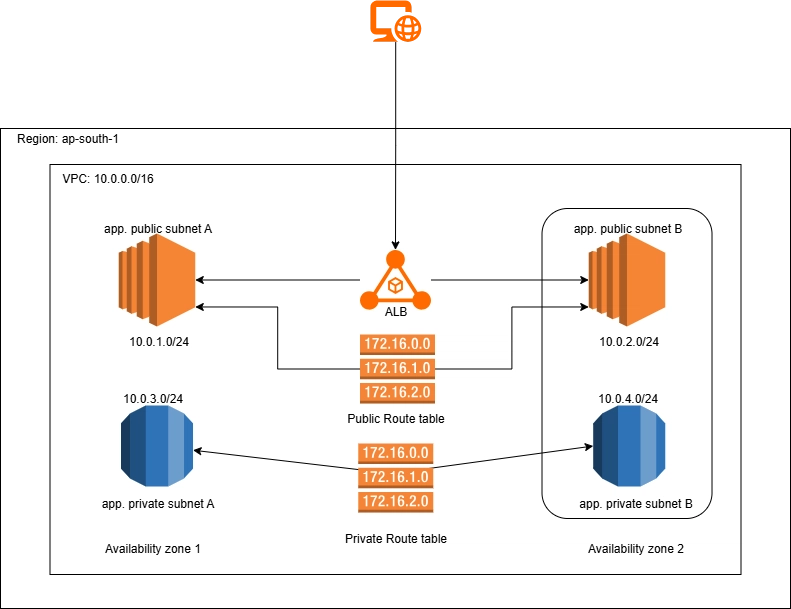
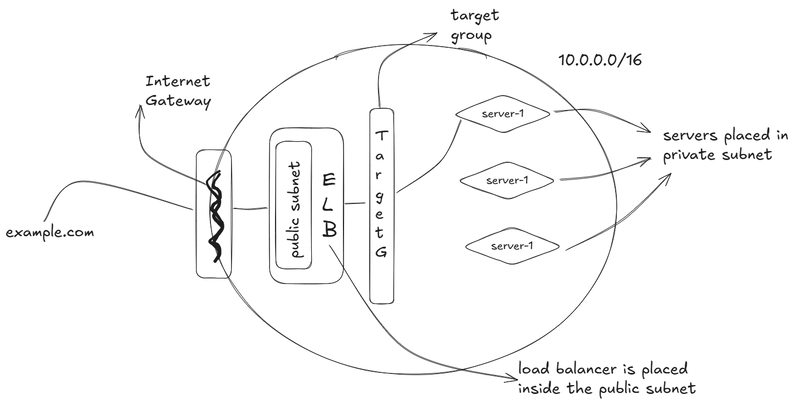
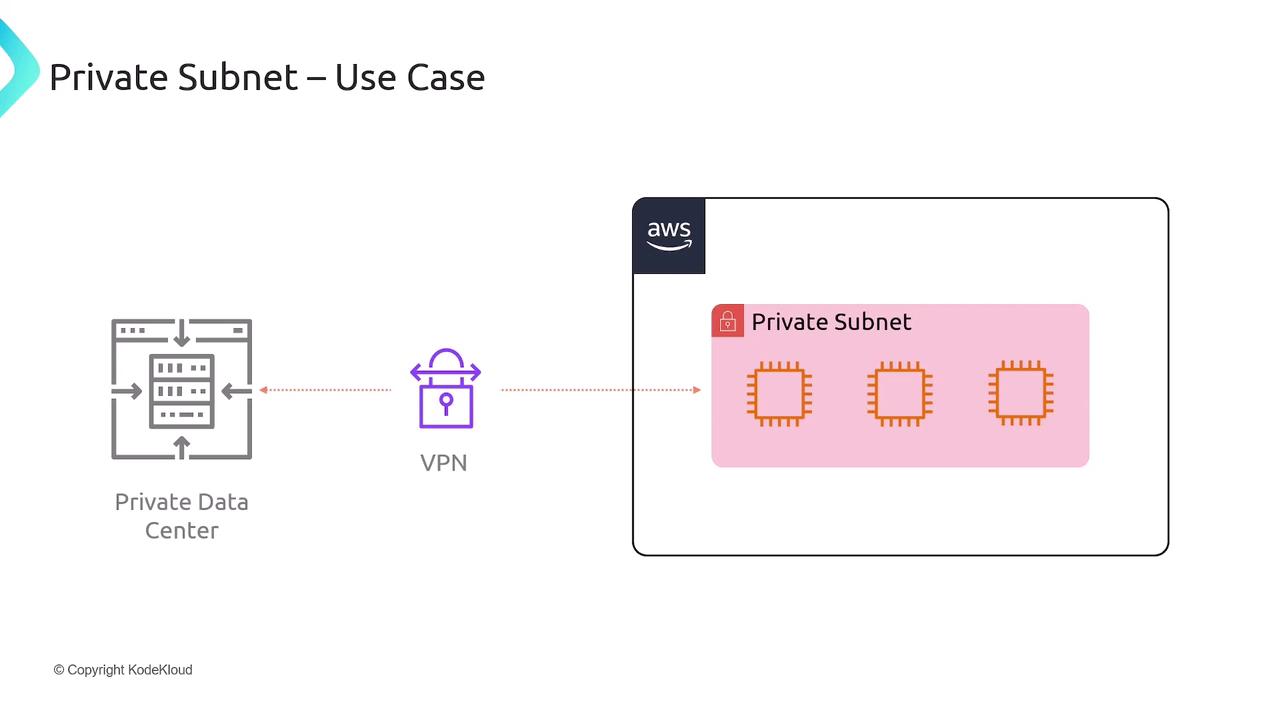
<https://aws.plainenglish.io/create-a-scalable-aws-vpc-with-auto-scaling-load-balancer-hands-on-learning-ce3abe9916fa>

<https://notes.kodekloud.com/docs/AWS-Networking-Fundamentals/Core-Networking-Services/Public-vs-Private-Subnets>









How to Choose Between Public and Private Subnets

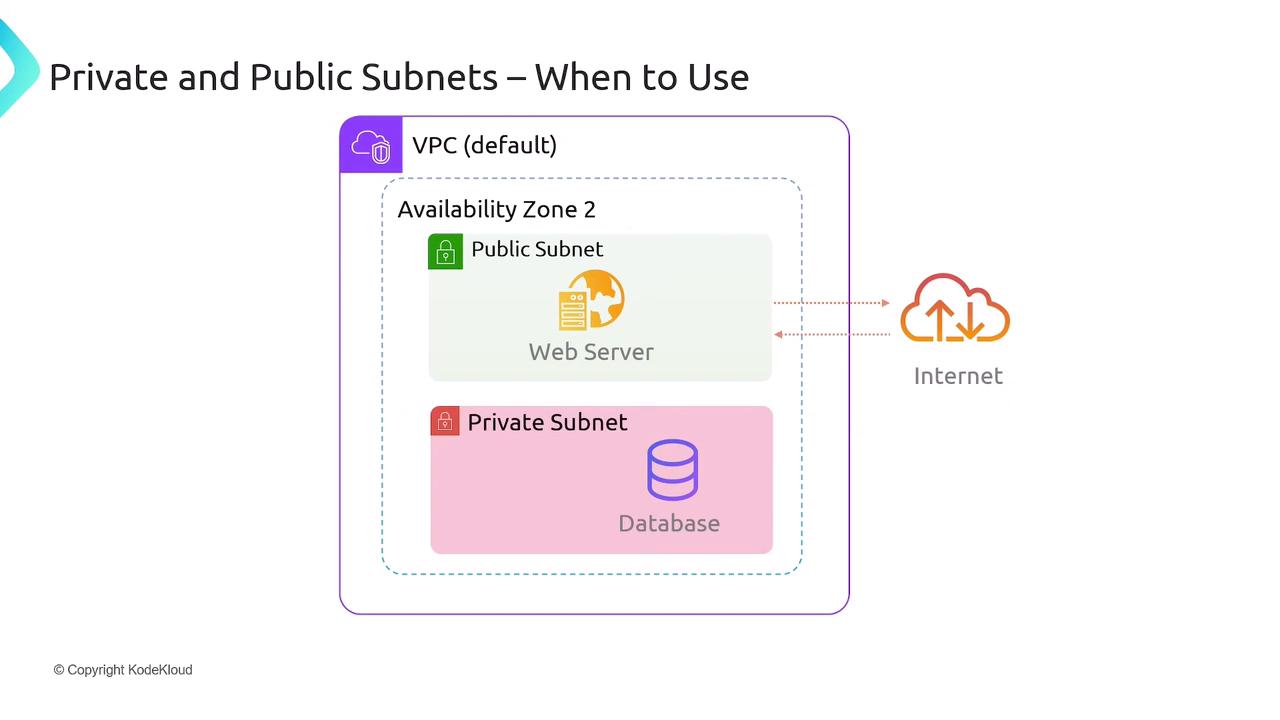
Ask yourself:

*Should resources in this subnet be directly accessible from the internet?*

* **Yes → Public subnet**
* **No → Private subnet**

Public Subnets

Public subnets host resources that must serve or receive traffic from the internet (for example, web servers, load balancers, or bastion hosts). These subnets have a route to an Internet Gateway (IGW) and a public IP assigned to each resource.



Key characteristics:

* **Route Table** includes a 0.0.0.0/0 route to the Internet Gateway
* **Inbound Traffic** allowed on specific ports (e.g., HTTP 80, HTTPS 443)
* **Outbound Traffic** to the internet for updates, API calls, etc.

Warning

Exposing resources in a public subnet means they are reachable over the internet. Always restrict inbound traffic with Security Groups and Network ACLs.

Private Subnets

Private subnets host internal resources (databases, application servers, caches) that should not be accessible directly from the internet. They typically use a NAT Gateway or NAT Instance in a public subnet for outbound internet access (software updates, API calls).

Key characteristics:

* **Route Table** has no direct route to IGW
* **Outbound Internet** enabled via NAT Gateway in a public subnet
* **No Public IPs** assigned by default

Typical Web Application Pattern

1. Clients connect to an Application Load Balancer or web server in a **public subnet**.
2. The web server in the public subnet communicates with a database in a **private subnet**.

This ensures:

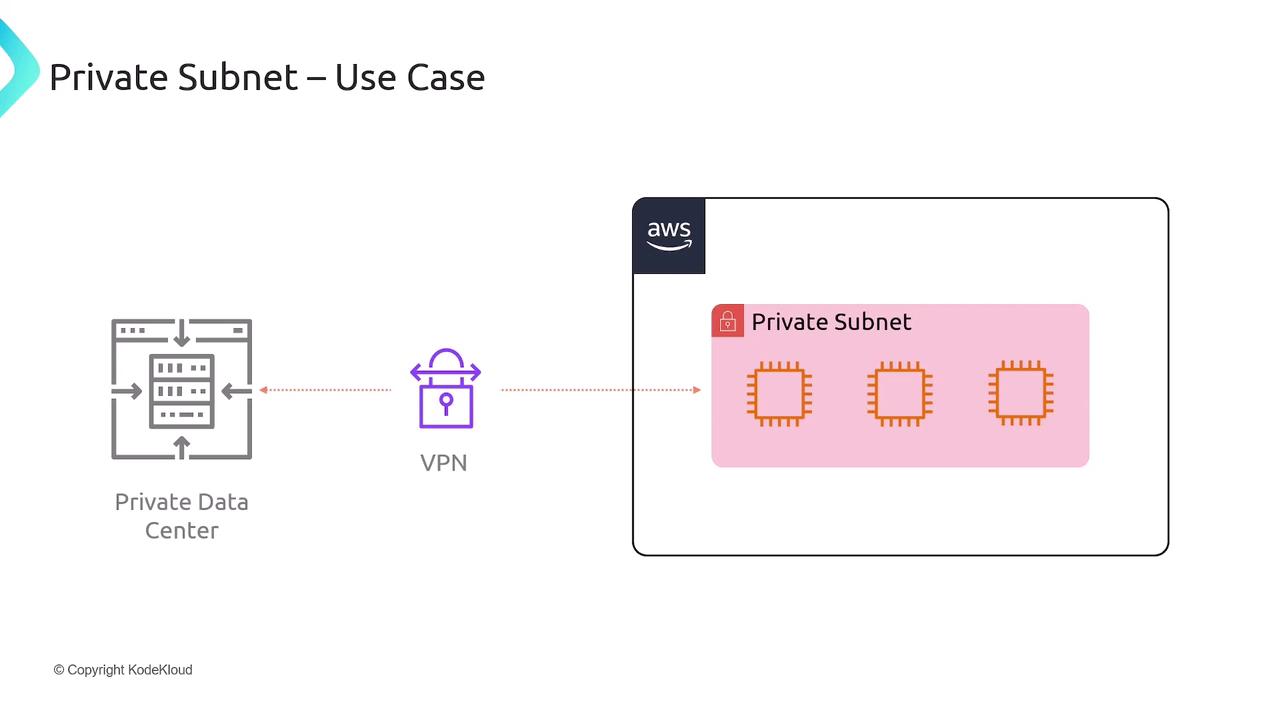
* End users never access the database directly.
* All data flows through controlled application logic.

Note

To allow your private subnet resources to fetch updates or reach external APIs, deploy a NAT Gateway in a public subnet and add the NAT’s Elastic IP to your private subnet’s route table.

Extending an On-Premises Data Center

Organizations often extend their corporate networks into AWS using VPN or AWS Direct Connect. In this architecture, all AWS resources reside in private subnets, and traffic flows over a secure tunnel to on-premises infrastructure.



Benefits:

* No need for Internet Gateway exposure
* Full integration with existing security controls
* Centralized network management

Subnet Comparison Table

| Feature | Public Subnet | Private Subnet |
| --- | --- | --- |
| Internet Connectivity | Direct via Internet Gateway | Indirect via NAT Gateway/Instance |
| Public IP Assignment | Yes | No |
| Typical Use Cases | Web servers, Load Balancers, Bastion Hosts | Databases, Application Servers, Caches |
| Ingress Traffic | Allowed from the internet (port-based controls) | Managed via Security Groups from within VPC |
| Egress Traffic | Allowed freely | Routed through NAT Gateway for outbound internet |

This guide covers the following topics:

**AWS VPC**  
-*Public Subnets  
-Internet Gateway  
-Route Table*

**AWS EC2**  
-*Autoscaling Group  
-Load Balancer  
-Target Group  
-Launch Template*

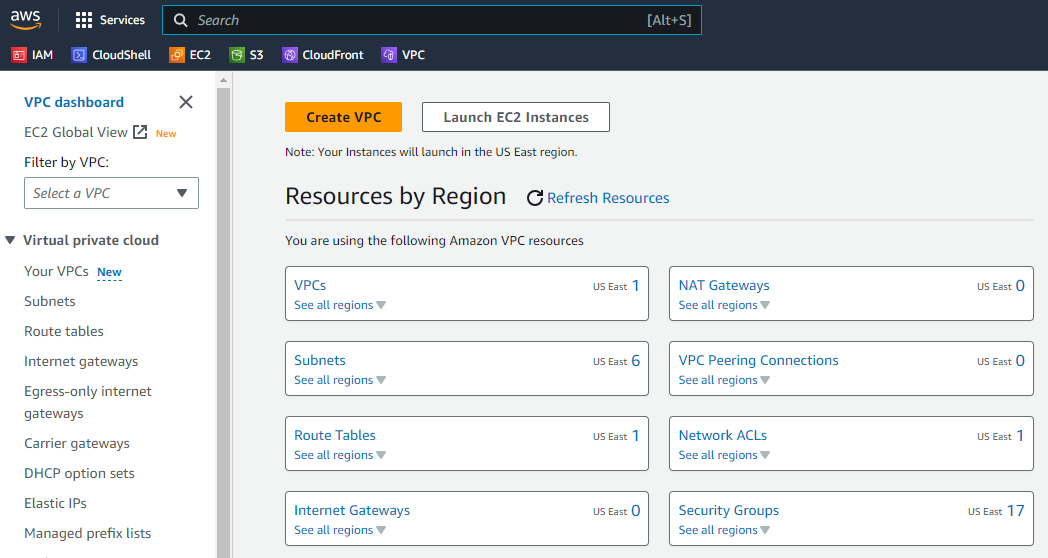
In the context of Level Up Bank’s progressive migration towards cloud infrastructure, we’re ready to explore a crucial step: setting up a Virtual Private Cloud (VPC) with three different subnets spread across separate Availability Zones (AZs). This guide will showcase how to create a VPC with multiple subnets spread across different availability zones and run a web server on an EC2 instance with auto-scaling and load-balancing capabilities.

**Creating the VPC:**

Let’s start off by laying the foundation by creating the VPC.

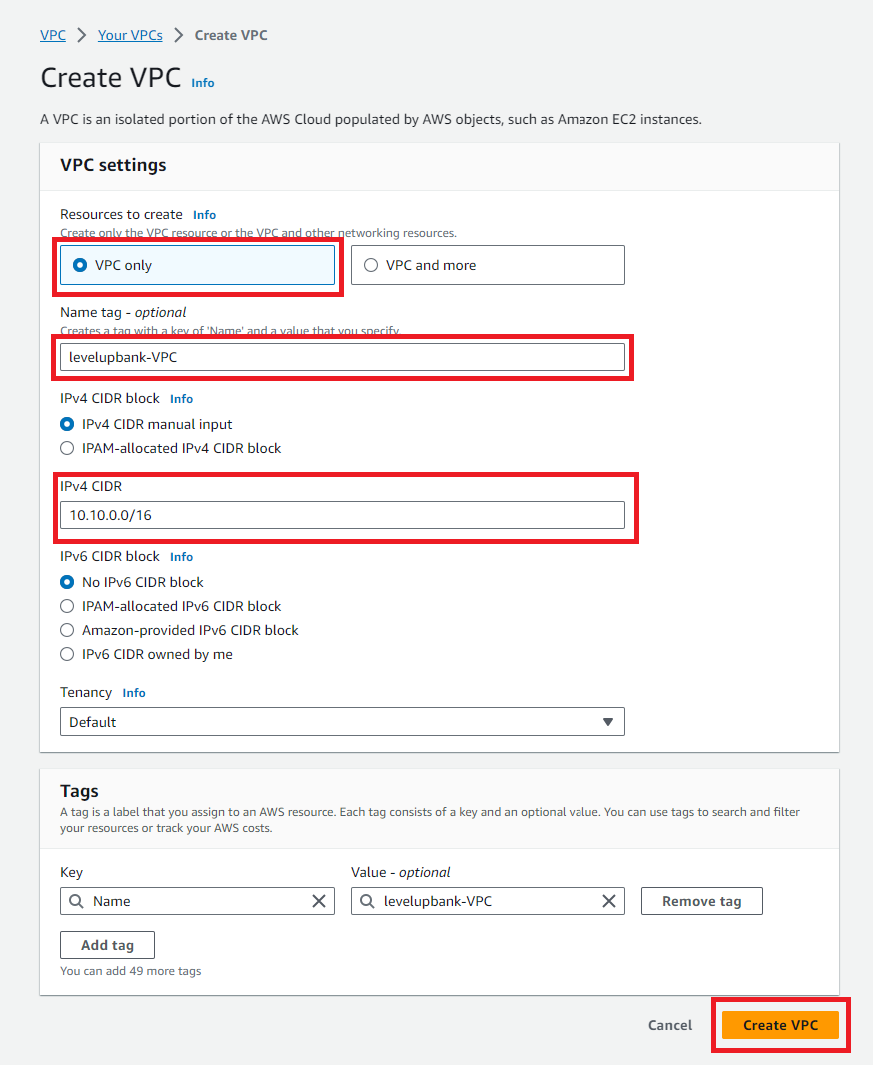
To do this, navigate to the AWS console and search for VPC.  
Click on “Create VPC” and provide a name for the VPC. In this case, it will be “levelupbank-VPC.”  
Set the CIDR block to 10.10.0.0/16, per Level Up Bank’s requirement. Leave other options as default, and create the VPC.

Press enter or click to view image in full size



Creating VPC — AWS

Press enter or click to view image in full size

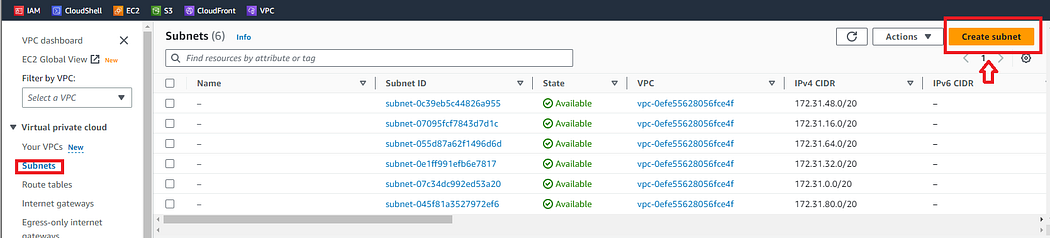


VPC Setup Settings

**Establishing Subnets:**

The next step is to establish the Subnets. In the VPC dashboard, select “Subnets” from the left-hand side menu.

Press enter or click to view image in full size

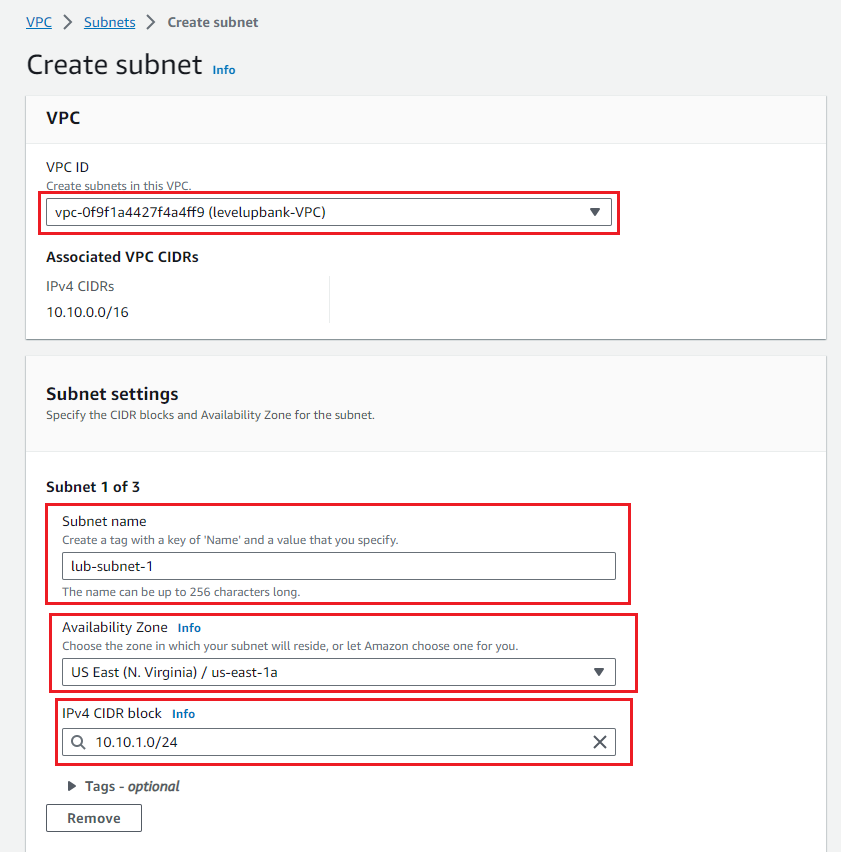


Create Subnet

Click on “Create subnet”  
Name the subnet. In this case, it will be — “lub-subnet-1” for the first one.  
Select the VPC — levelupbank-VPC,  
Select an availability zone  
Set the CIDR block to 10.10.1.0/24.

Click on “Add new subnet” to create 2 more subnets.

Press enter or click to view image in full size



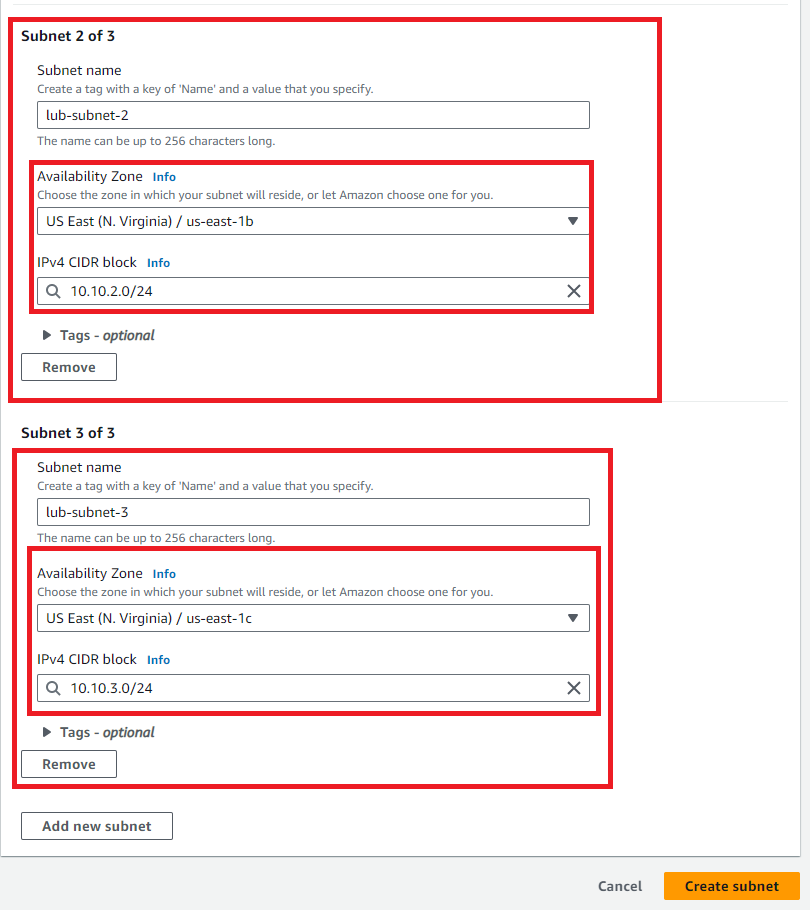
Creating Subnet 1

Subnet 2: Repeat the process for the second subnet with CIDR block 10.10.2.0/24, a different name and a different availability zone.

Subnet 3: Repeat the process for the second subnet with CIDR block 10.10.3.0/24, a different name and a different availability zone.

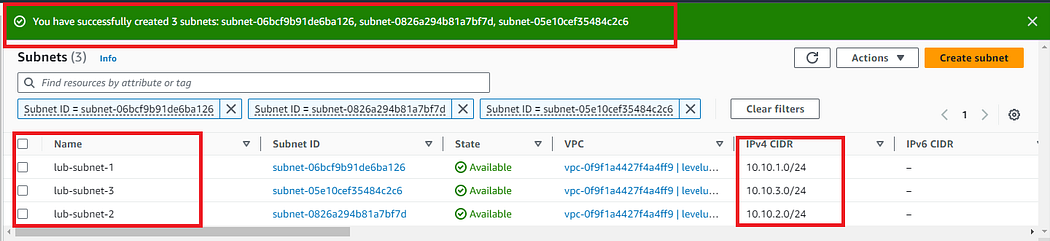
Note that each subnet should be in a different availability zone to increase fault tolerance and ensure high availability.

Press enter or click to view image in full size



Creating Subnets 2 & 3

Press enter or click to view image in full size

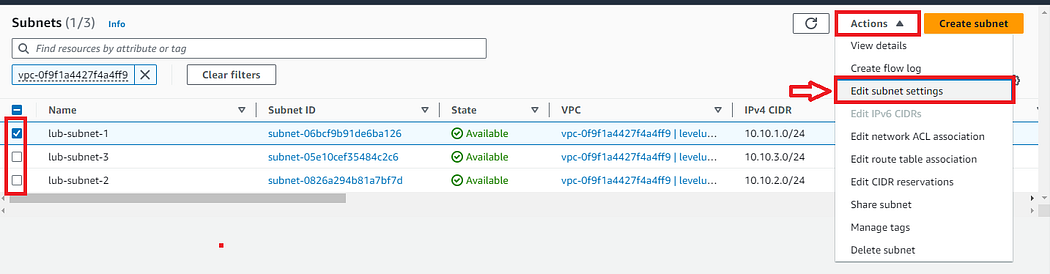


Subnet Creation Success

Once the subnets are created, enable the auto-assign IP option to automatically assign public IPs to the EC2 instances that will be launched automatically with Auto Scaling.

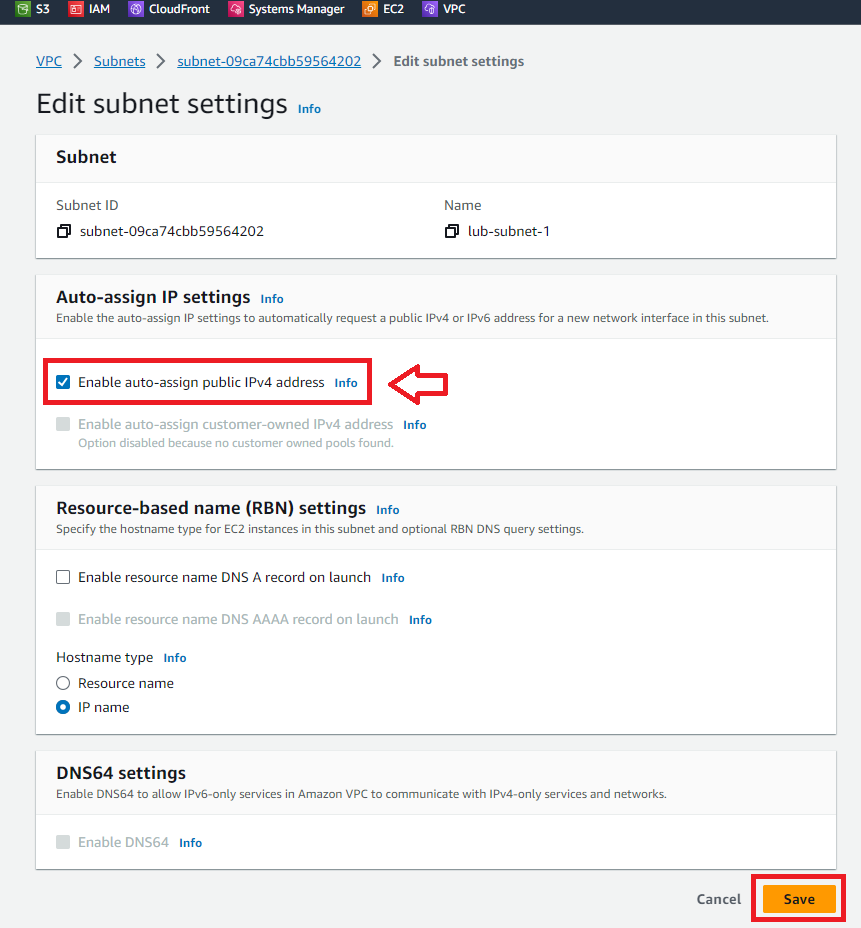
On the Subnet menu page, select one subnet, click on “Edit subnet settings,” and ensure that “Enable auto-assign public IPv4 address” is enabled. Do this for all 3 subnets.

Press enter or click to view image in full size



Editing subset settings — VPC

Press enter or click to view image in full size



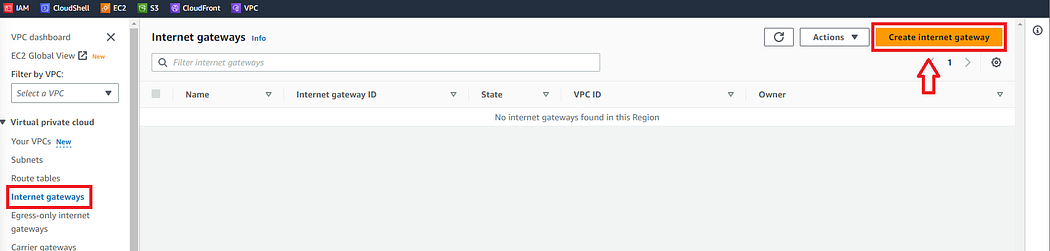
Enabling Auto-assign IPv4 — Subnet Settings

**Creating an Internet Gateway:**

After creating & configuring the subnets in the VPC, we can create an Internet Gateway to allow internet connection to our VPC.

From the VPC left-hand side menu, select Internet Gateway and click on “Create Internet Gateway.” Name the Internet Gateway and click on Create.

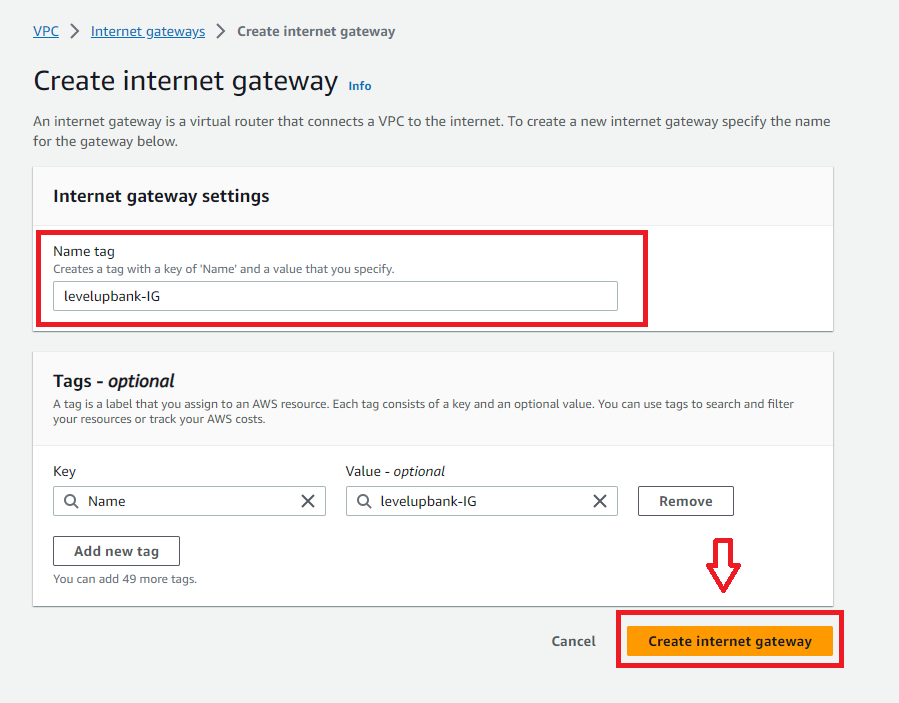
Press enter or click to view image in full size



Create Internet Gateway

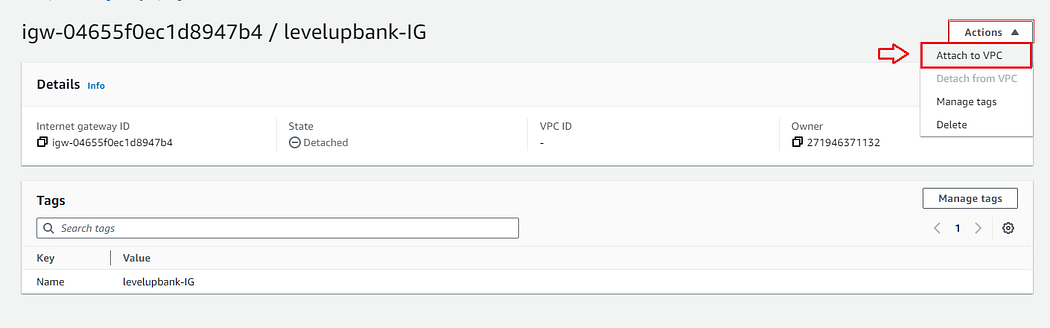
Once created, we need to attach it to our desired VPC, which in this case is Level Up Bank’s VPC. Click on the “Actions” drop-down menu and select “Attach to VPC.” Select the VPC and click on “Attach internet gateway.”

Press enter or click to view image in full size



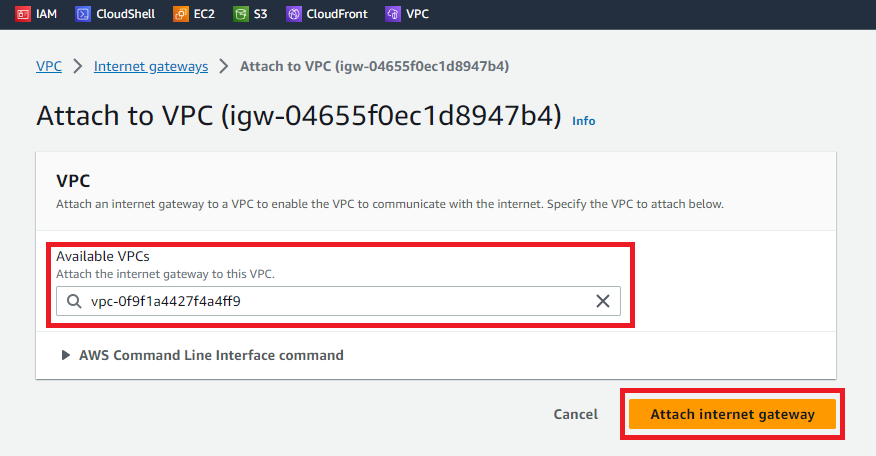
Creating the Internet Gateway

Press enter or click to view image in full size



Attaching VPC to Internet Gateway

Press enter or click to view image in full size



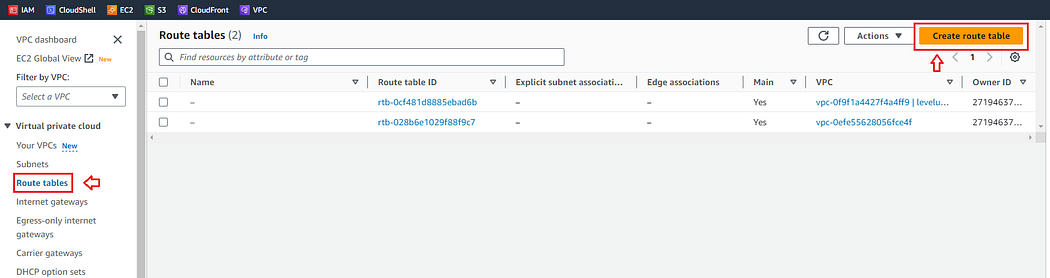
Attaching VPC to Internet Gateway

**Creating the Route Table:**

Moving forward to Route Tables. Route tables are like navigation guides for network traffic, showing the best paths between different parts of the virtual network, such as subnets and the internet. They ensure data reaches the correct destinations, making communication efficient and organized within the VPC.

To create a route table, click on “Route Table” from the VPC left-hand side menu.

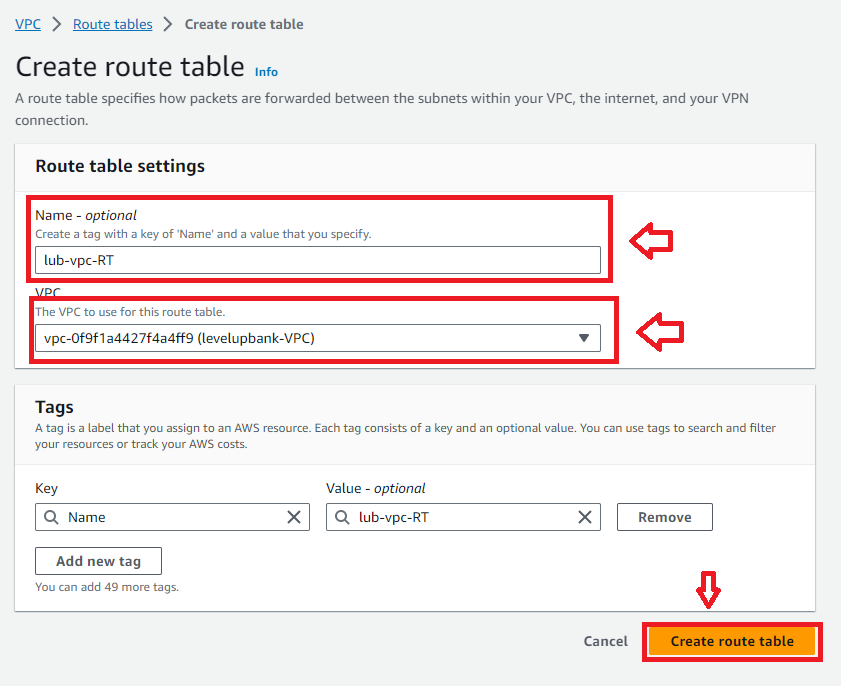
Press enter or click to view image in full size



Create Route Table

Name the route table on the Create Route Table menu, select the VPC, and click “Create route table.”

Press enter or click to view image in full size

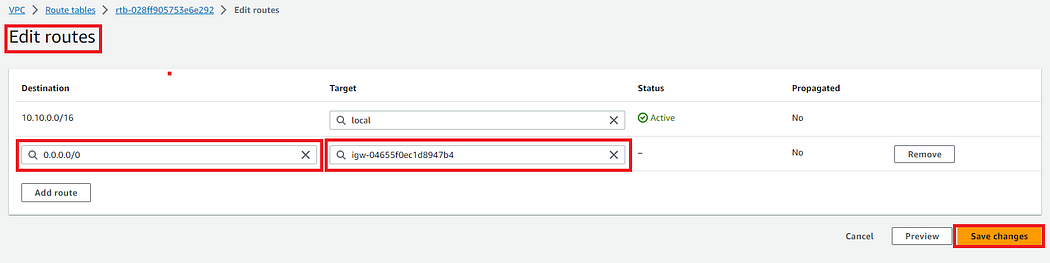


Creating Route Table — Configurations

Once the route table has been created, we must do two things. One is to configure it so that it routes the network traffic to our VPC.

The first thing, on the route table page, click on “Edit routes.” Click on “Add route,” add the destination “0.0.0.0/0”, select the VPC for the “Target,” and click “Save changes.”

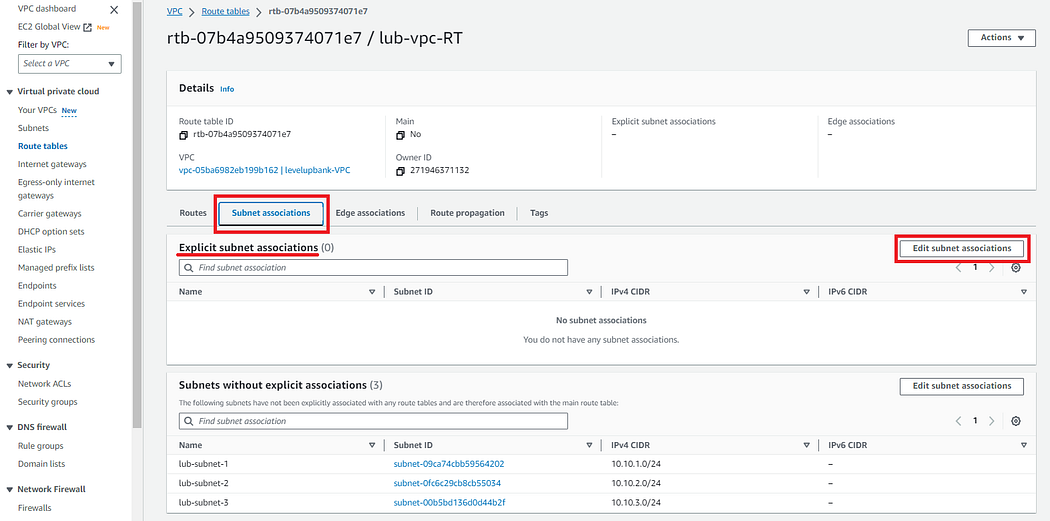
Press enter or click to view image in full size



Adding Internet Gateway to Route Table

The second thing, click on Subnet associations and edit Explicit subnet associations.

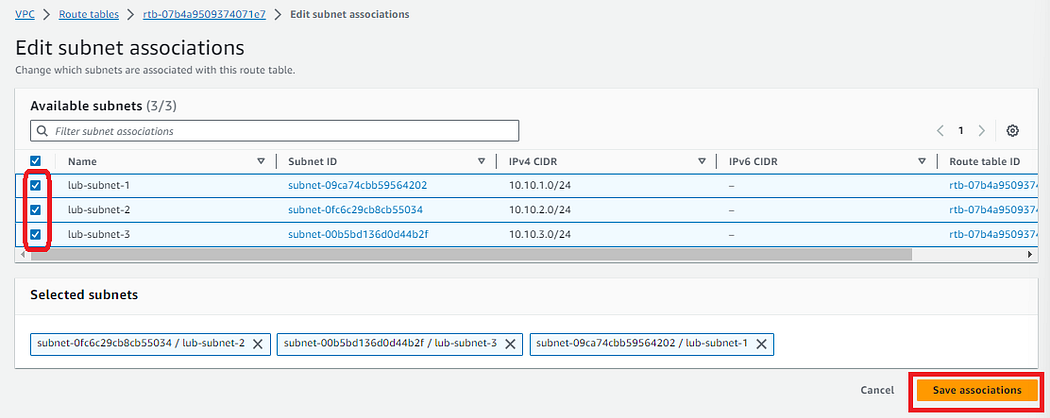
Press enter or click to view image in full size



Route Table Subnet Associations

Select the three subnets and Save associations.

Press enter or click to view image in full size



That’s it for VPC Configurations and setting up everything on this end. Now let's move on to EC2.

Get Khalil Faqiri’s stories in your inbox

Join Medium for free to get updates from this writer.

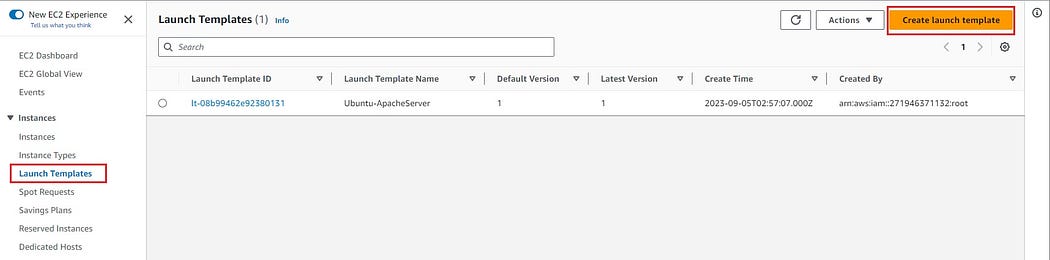
Subscribe

Moving on to EC2 Instance — The auto-scaling group will create the EC2 instances based on a launch template. This launch template will specify our custom AMI tailored for Level Up Bank’s Server, which is an AMI running an Ubuntu image with Apache installed and enabled.

**Creating a Launch Template:**

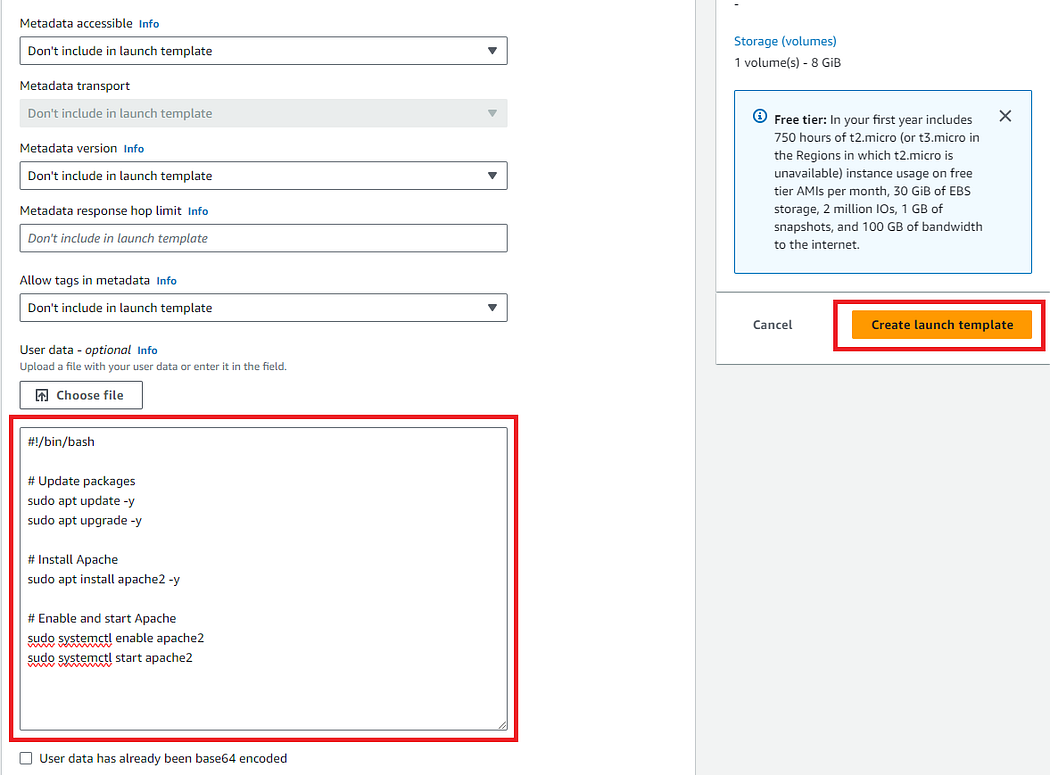
To create a launch template, navigate to the EC2 dashboard and locate “Launch Template” from the left-hand side menu.

Press enter or click to view image in full size



On the launch template page:  
- Click on “Create launch template.”  
- Name the template.  
- Choose the OS image.  
- Instance Type.  
- Select a key pair — this will allow us to access it via SSH later.  
- Network Setting — Security group allowing SSH & HTTP access.  
- Finally, under the “advanced details” section, write/paste your bash script, and it will execute as soon as the server is launched.  
- Leave other settings on default and create the launch template

Press enter or click to view image in full size



Launch Template — Bash Script

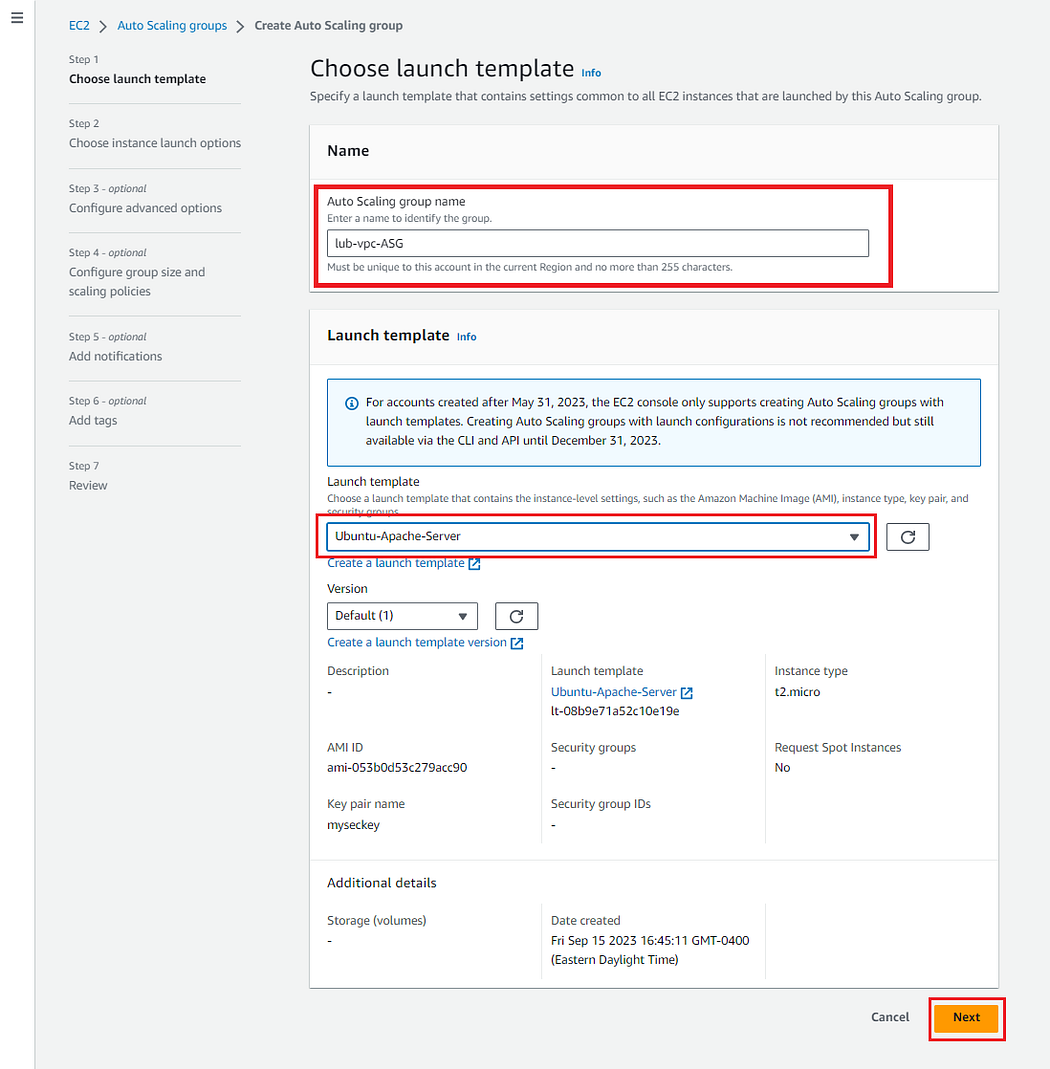
With the launch template ready to be used, we can head to Auto Scaling Groups to start launching instances.

**Creating Auto Scaling Group:**

Navigate to EC2 left-hand side menu and select Auto Scaling Groups, click on Create Auto Scaling group.

Pick a name for the Auto Scaling Group and select the desired launch template, which is the Ubuntu image running the Apache server in this case. Click Next.

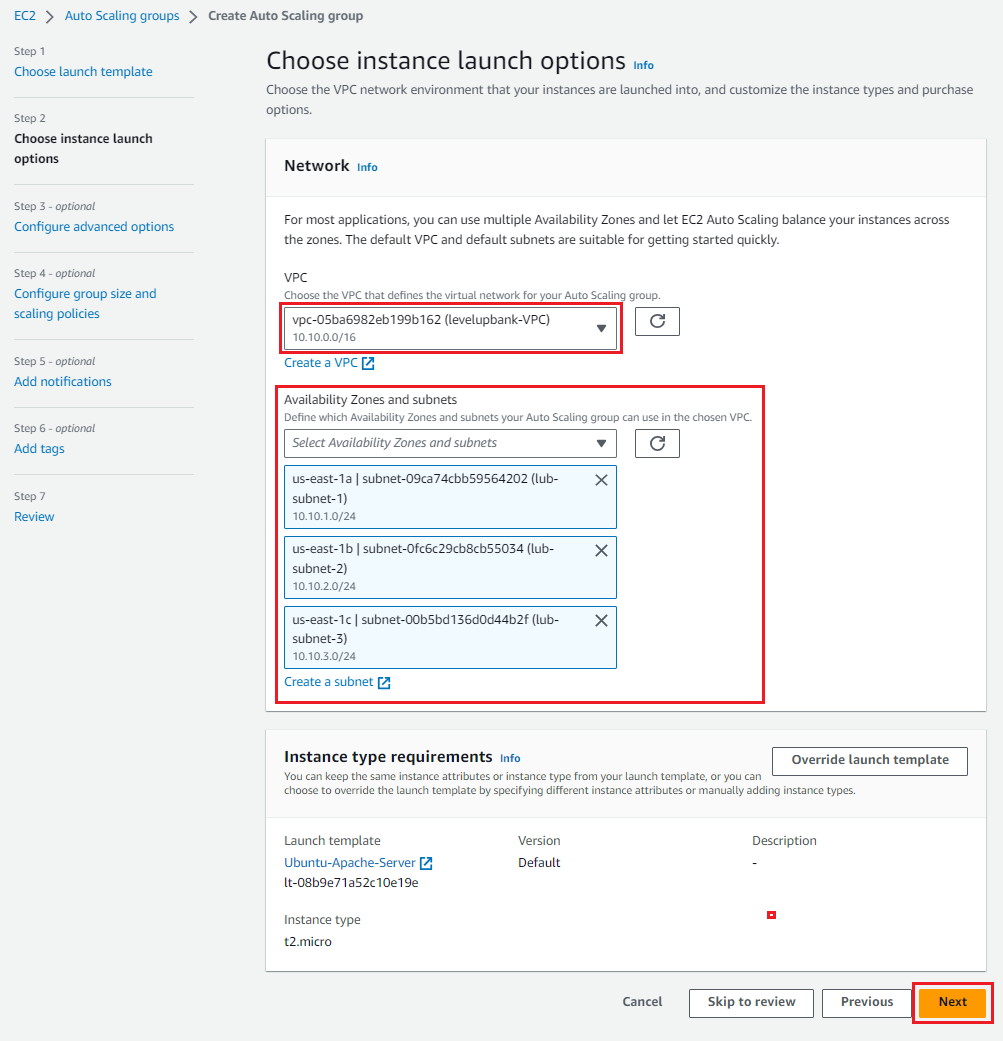
Press enter or click to view image in full size



ASG — Choosing Name & Launch Template

Select the desired VPC. In this case, it is the levelupbank-VPC.  
Select the three different availability zones in which we launched the subnets. All of them should be visible under the drop-down menu. Click Next.

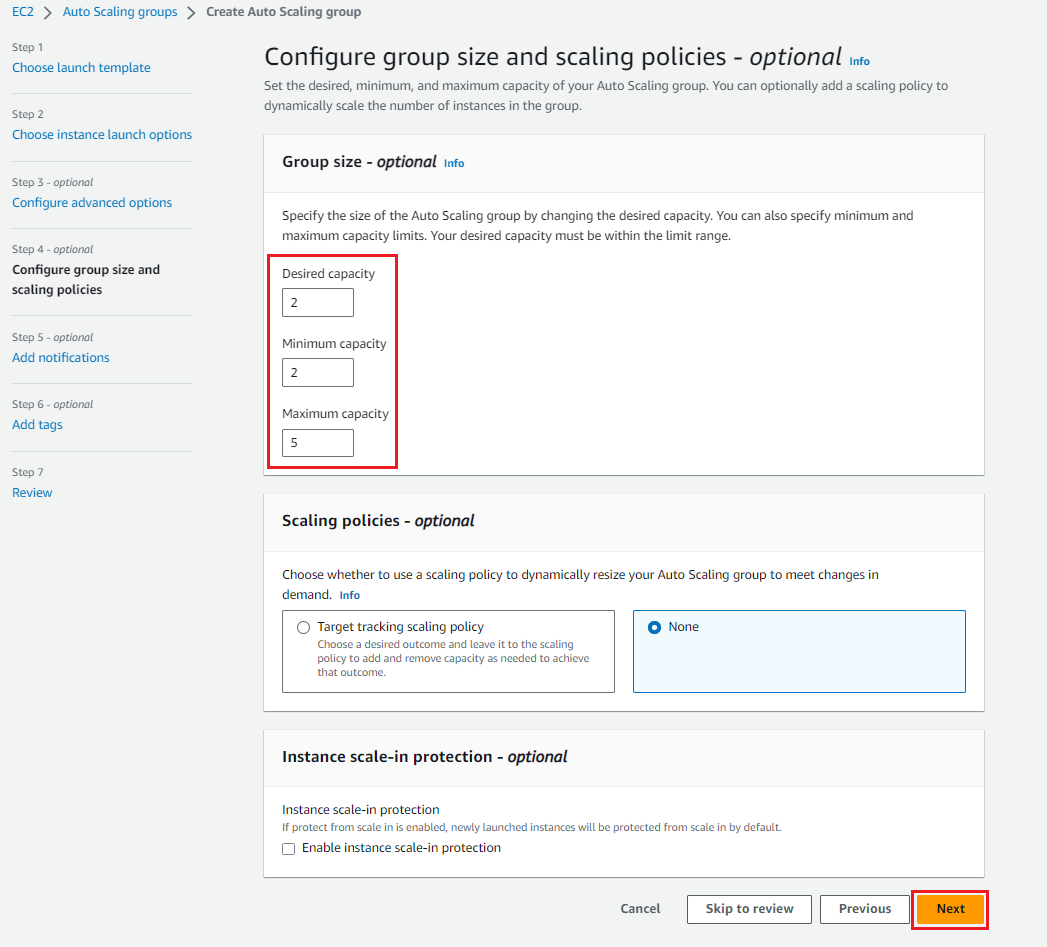
Press enter or click to view image in full size



ASG — Choosing VPC & AZs

Next, we declare the desired, minimum, and maximum capacity. This determines the number of instances launched. This number can vary depending on project requirements. Input your values and click on Next.

Press enter or click to view image in full size



ASG — Capacity & Gorup Size

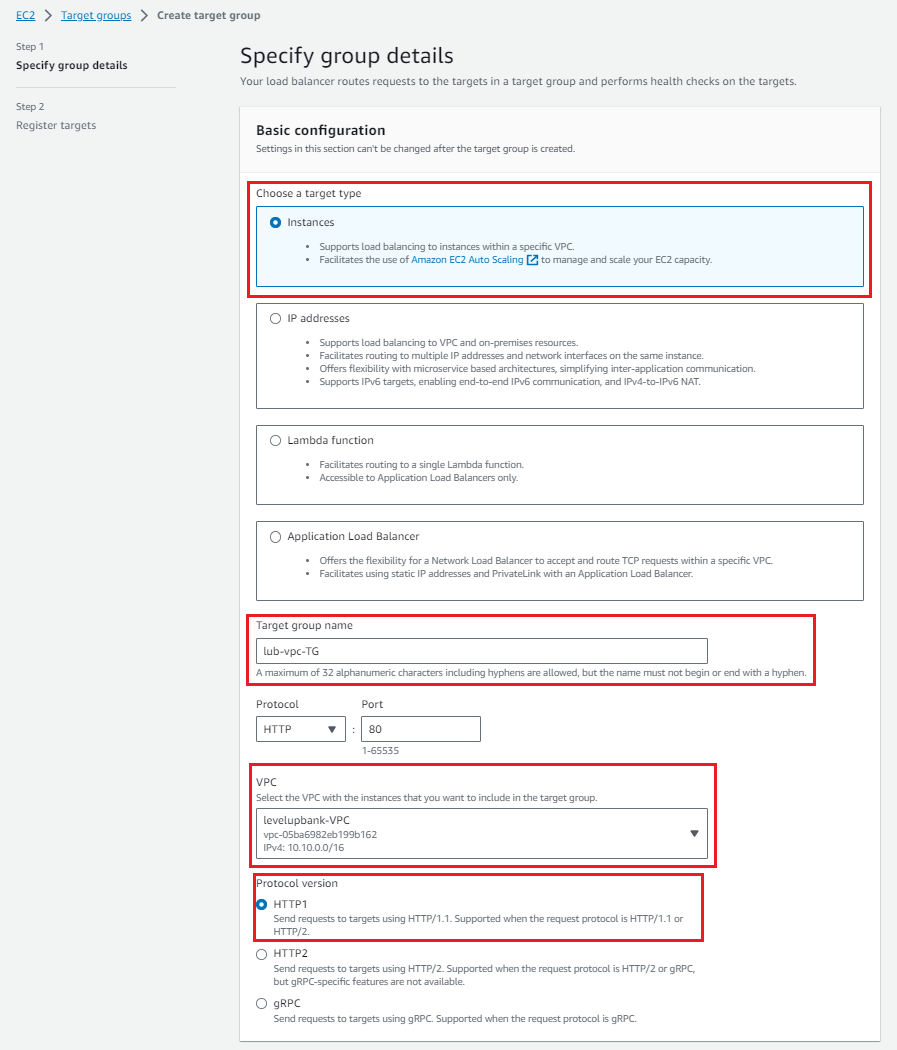
Leave other settings on default, review everything, and finally, create the Auto Scaling Group.

From here, we move on to creating a Load Balancer to distribute traffic between instances. But before we get to that, we must quickly create Target Groups.

**Creating Target Groups**

On the EC2 dashboard, find and click on Target Groups. Select “Create target group”  
Select the target type, “Instances” in this case.  
Name the target group.  
Select the VPC and ensure that HTTP1 is selected since it is a web server.  
Click Next.

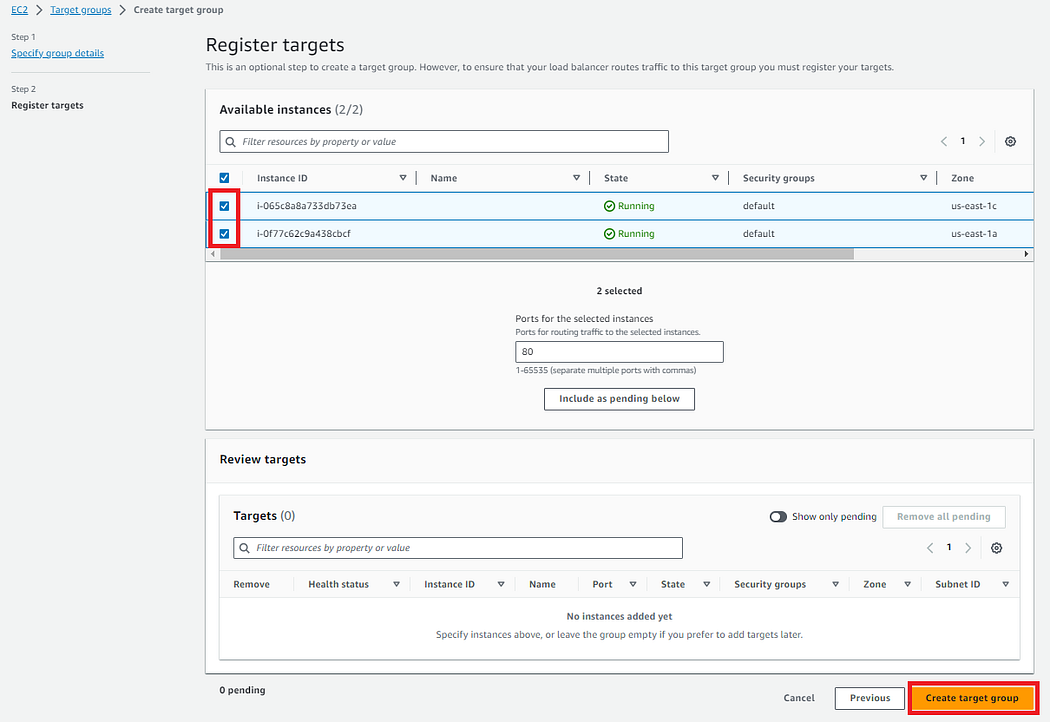
Press enter or click to view image in full size



Creating a Target Group — Configuration

Select the available instances and create the target group.

Press enter or click to view image in full size

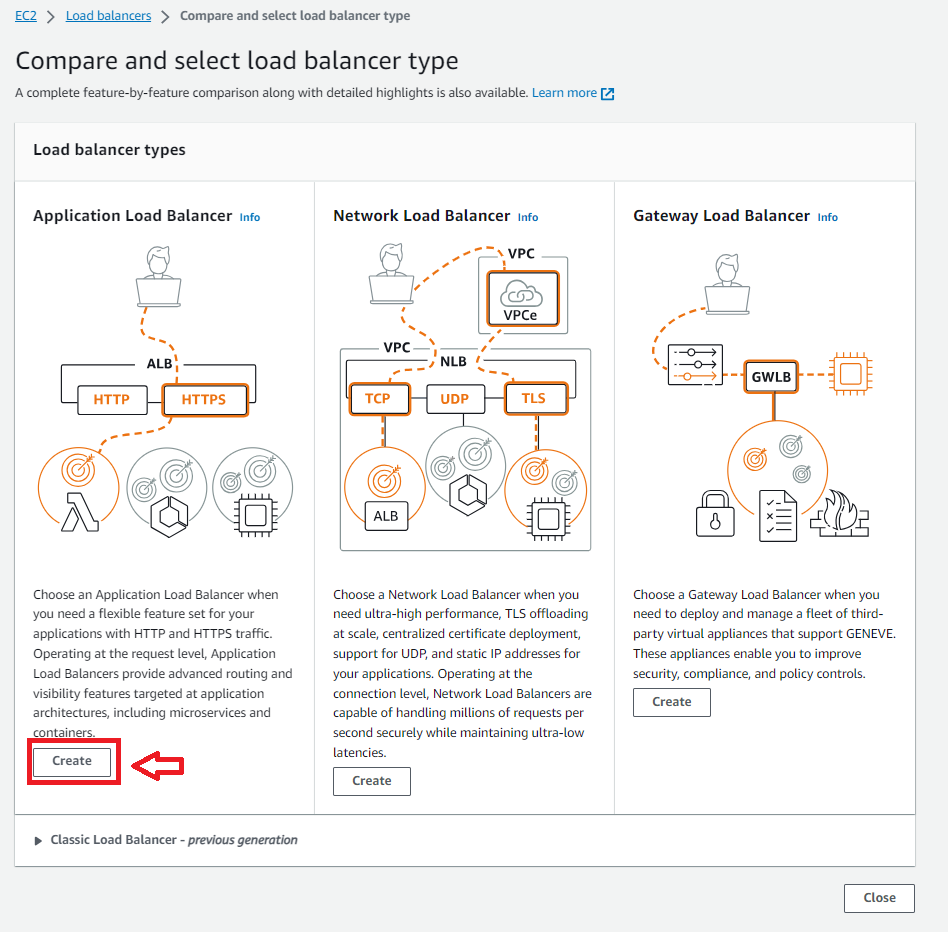


Once the target group is successfully created, we move on to creating the Load Balancer.

**Creating the Load Balancer:**

On the EC2 dashboard, find and click on Load Balancers. Select “Create load balancer”  
Select “Application Load Balancer” for the balancer type.

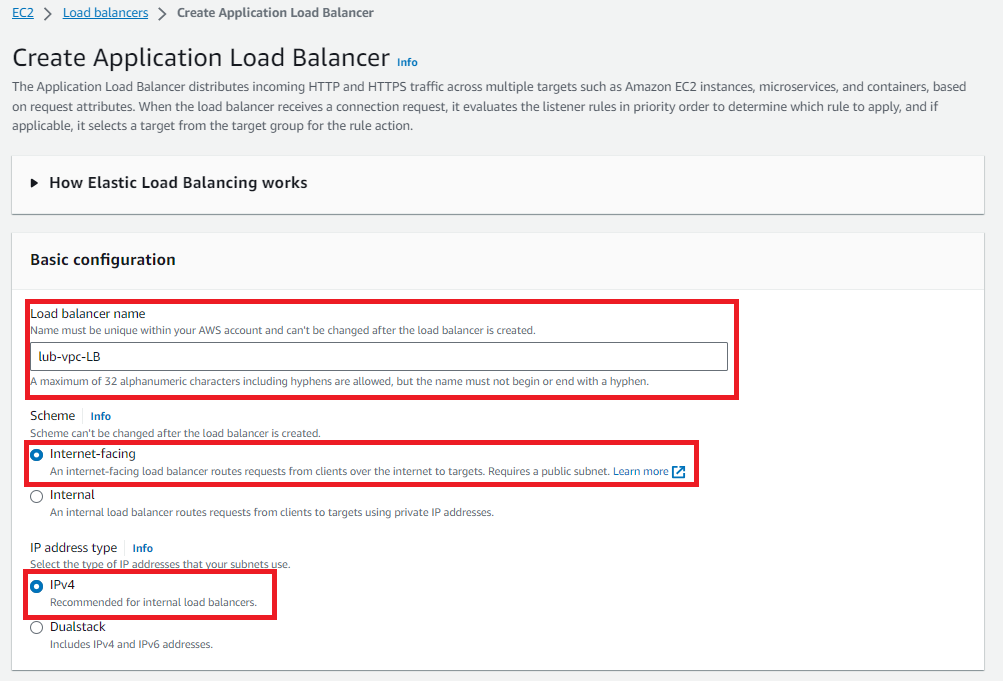
Press enter or click to view image in full size



Creating Load Balancer — Selecting Balancer Type

- Name the load balancer  
- Select “Internet-facing” scheme  
- Select IPv4 for the IP address type

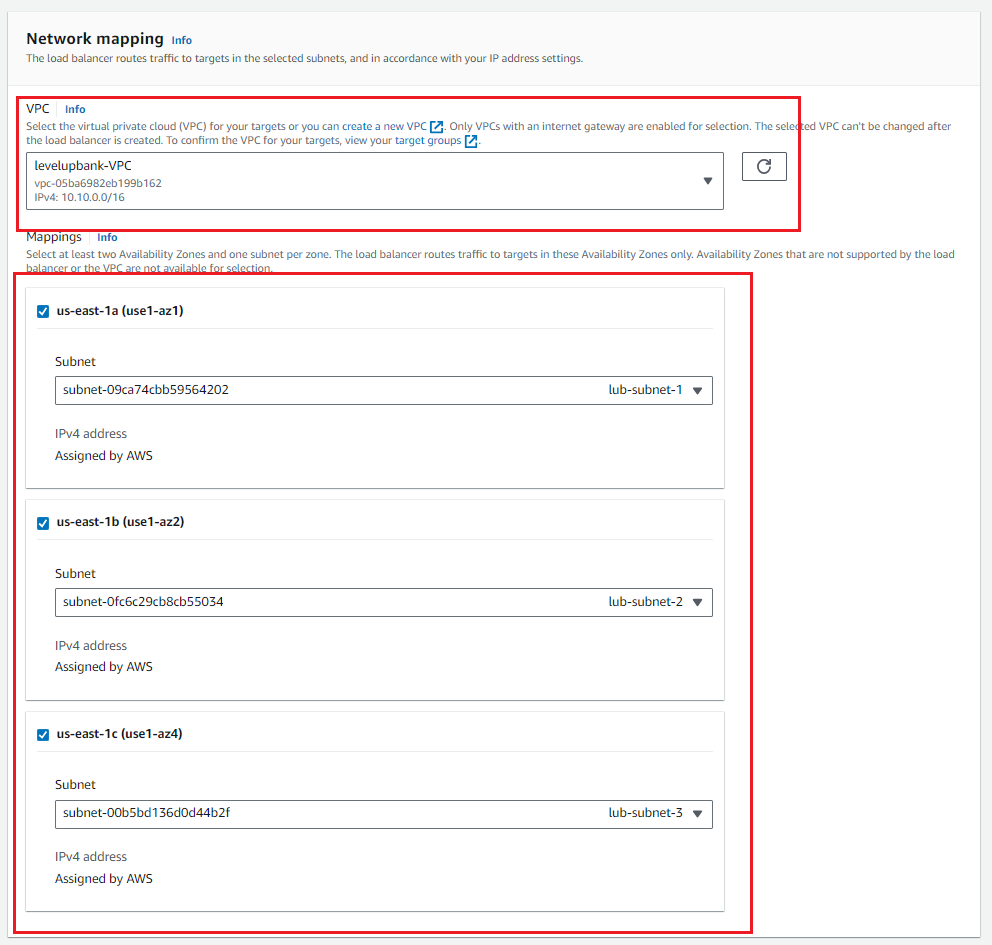
Press enter or click to view image in full size



Load Balancer — Basic Config

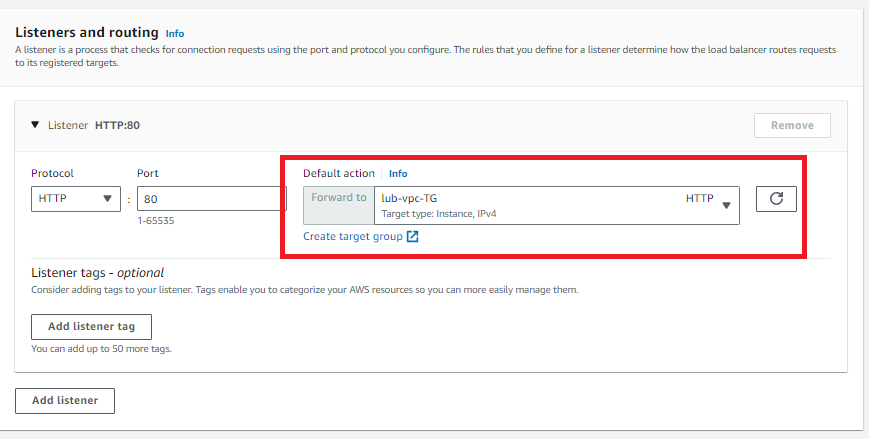
- Select the VPC under Network mapping  
- Select all three subnets  
- Select Target group under Listeners and routing.  
- Finally, review & create the load balancer.

Press enter or click to view image in full size



Load Balancer — Network Mapping

Press enter or click to view image in full size

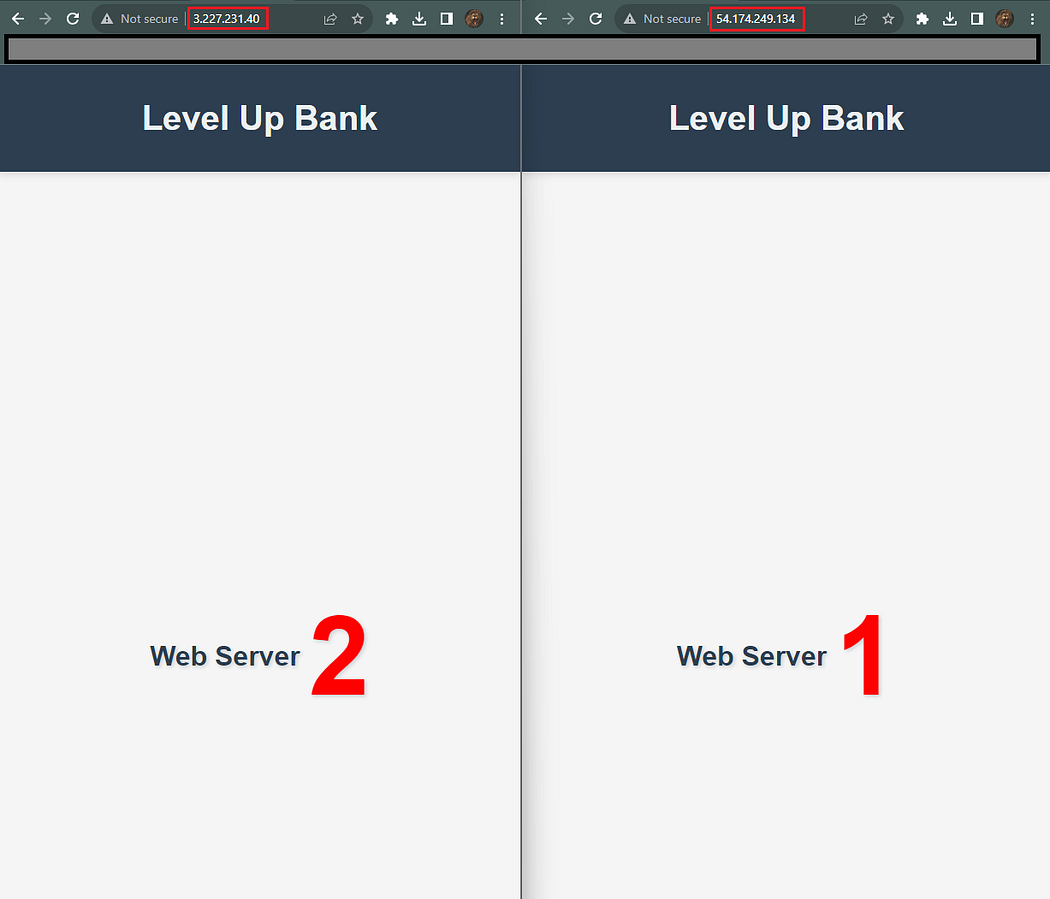


Load Balancer — Target Group

Testing Setup:

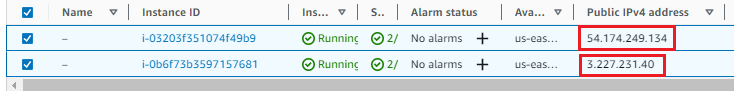
It is time to put our setup to the test and see if we can access the webserver. The web server can be accessed using the public IPv4 address of the EC2 instance.

Press enter or click to view image in full size



Level Up Bank’s Servers

Press enter or click to view image in full size



Ec2 Instance IPv4 Addresses

Voilà! We’ve successfully set up a robust VPC that’s both scalable and secure. While this might seem basic, with AWS’s incredible capabilities, it’s geared up to handle immense workloads. Let’s harness its power!

Thank you for reading my article! If you found it insightful, follow me for more projects!