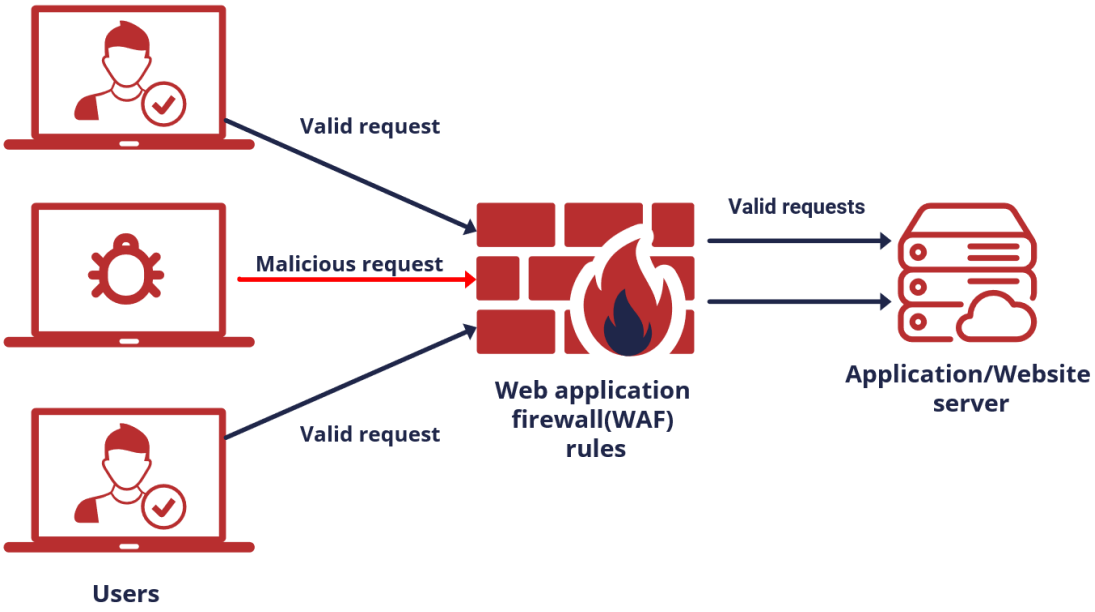
Securing a Web Application with AWS WAF



This document outlines a project to deploy a basic web server on an AWS EC2 instance and protect it using an Application Load Balancer (ALB) and AWS Web Application Firewall (WAF). The project demonstrates fundamental AWS networking concepts and security best practices by creating a custom Virtual Private Cloud (VPC), launching a web server, and configuring WAF to filter malicious traffic. The goal is to set up a robust, scalable, and secure web application infrastructure.

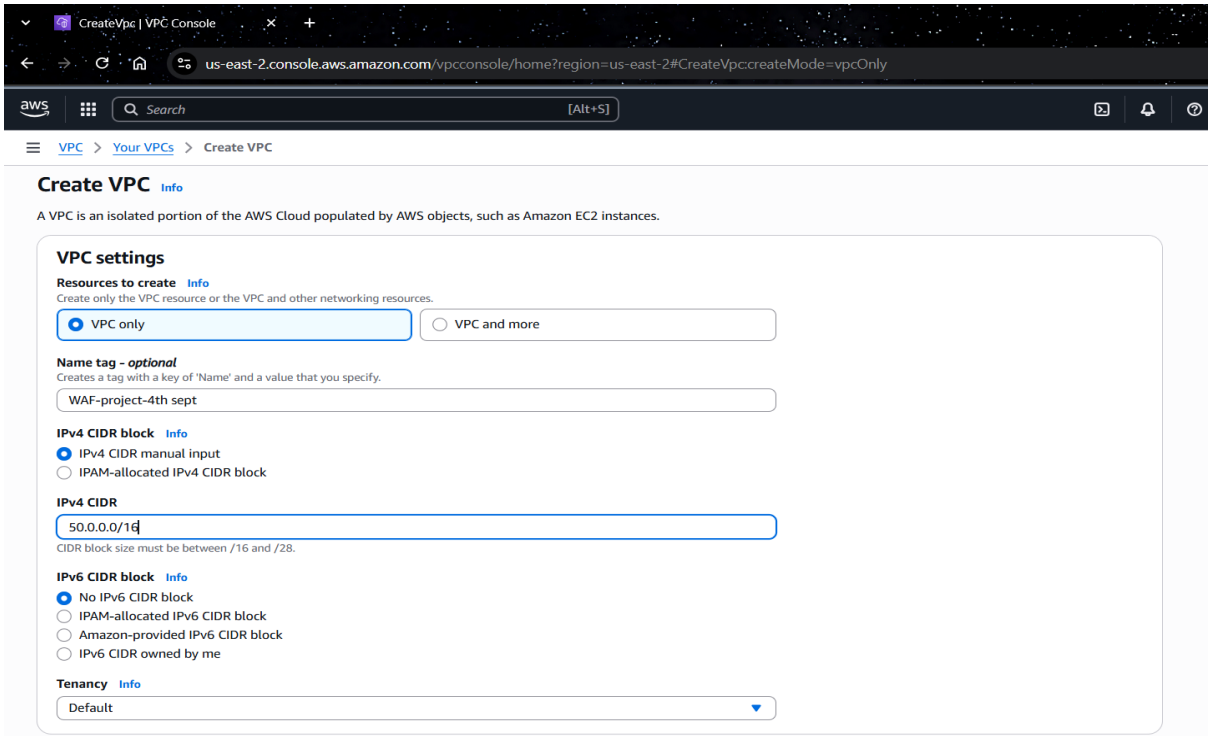
- **What is AWS WAF?** AWS Web Application Firewall (WAF) helps protect your web applications or APIs against common web exploits and bots that may affect availability, compromise security, or consume excessive resources. AWS WAF gives you control over which traffic to allow or block to your web applications by defining customizable web access control lists (web ACLs), rules, and rule groups.

Step-by-Step Guide to Deploying the Application

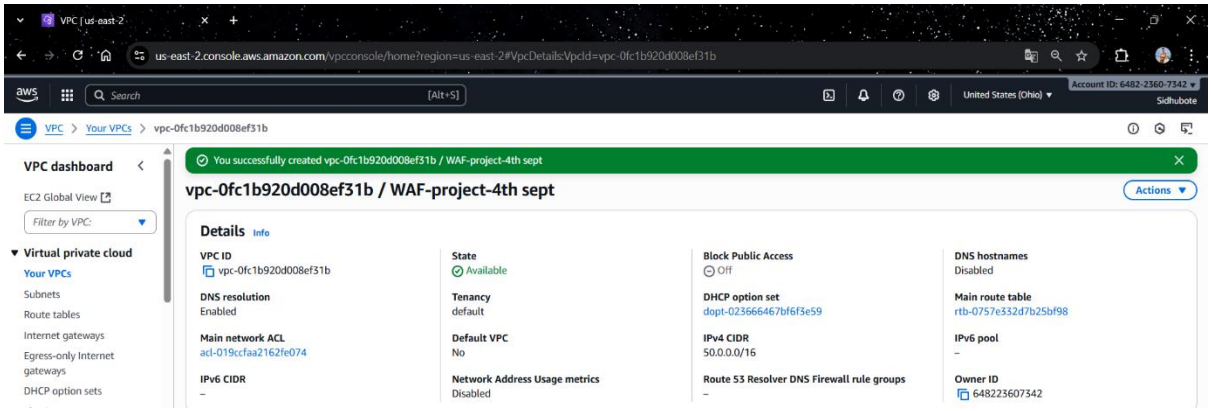
1. VPC and Networking Setup

First, you need to set up the network for your application. This involves creating a custom VPC, subnets, an Internet Gateway, and a route table.

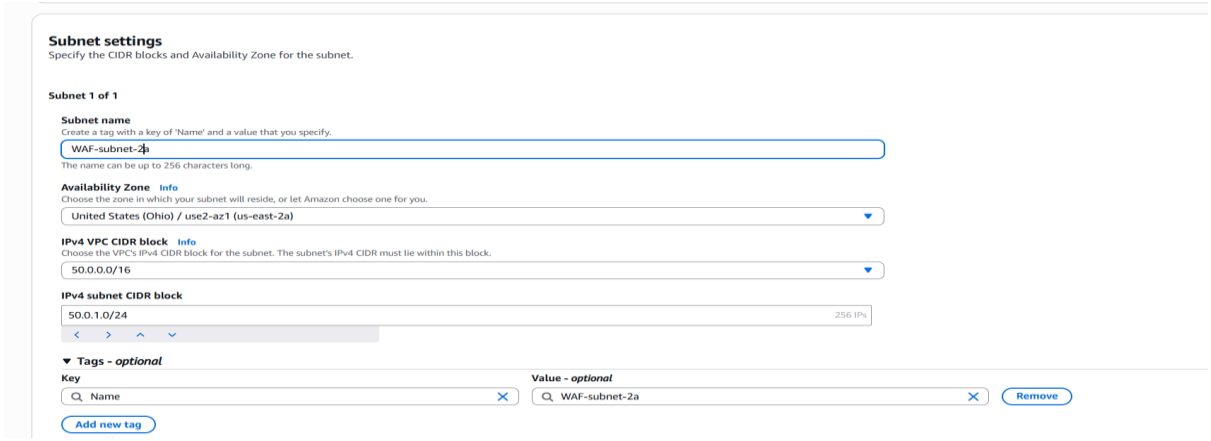
- **Create VPC**: Navigate to the VPC dashboard and create a new VPC. Give it a name like WAF-project-4th sept and a CIDR block (e.g., 50.0.0.0/16).



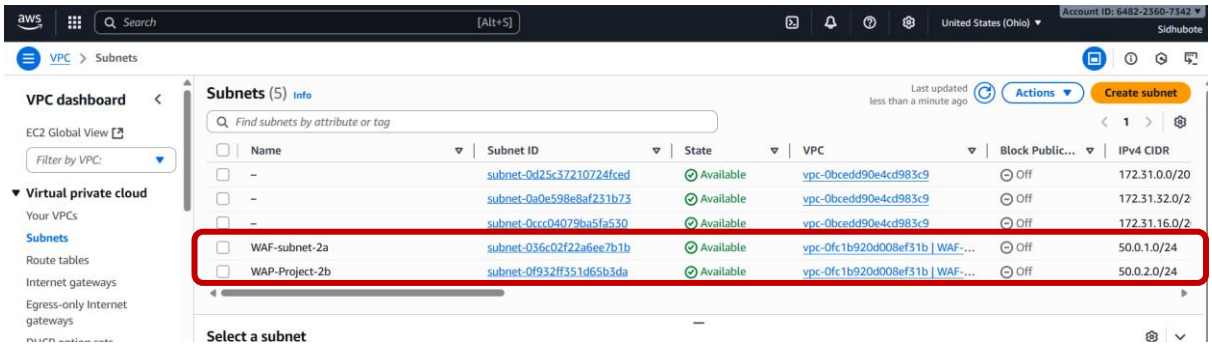
- **Verify VPC**



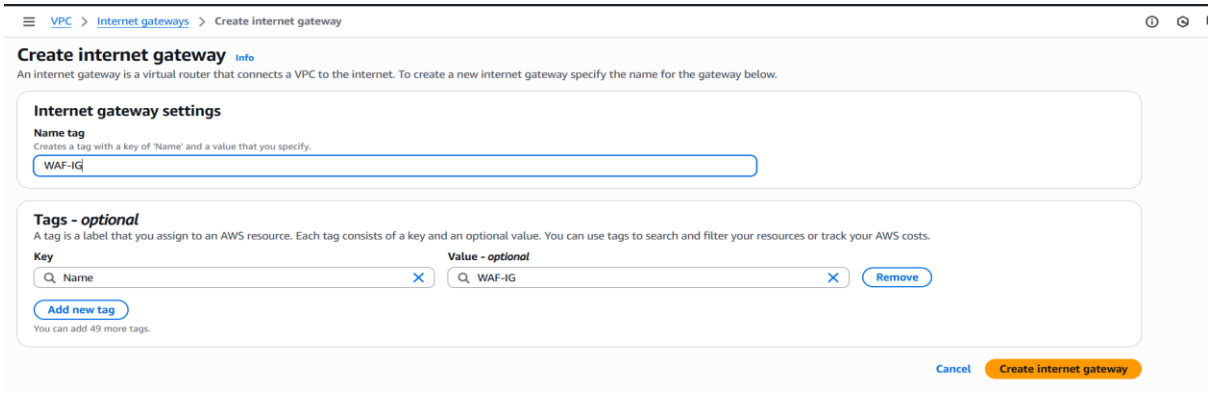
- **Create Subnets**: Create two subnets within the VPC you just created. For example, WAF-subnet-2a with a CIDR block of 50.0.1.0/24 and WAF-Project-2b with a CIDR block of 50.0.2.0/24.



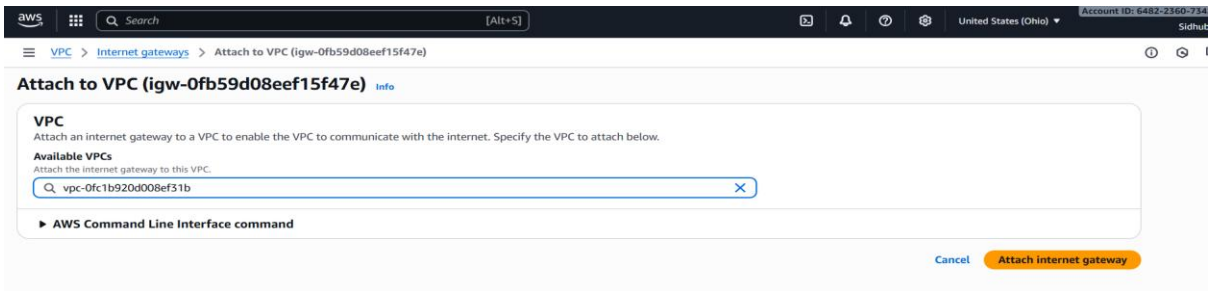
- **Verify the subnet**



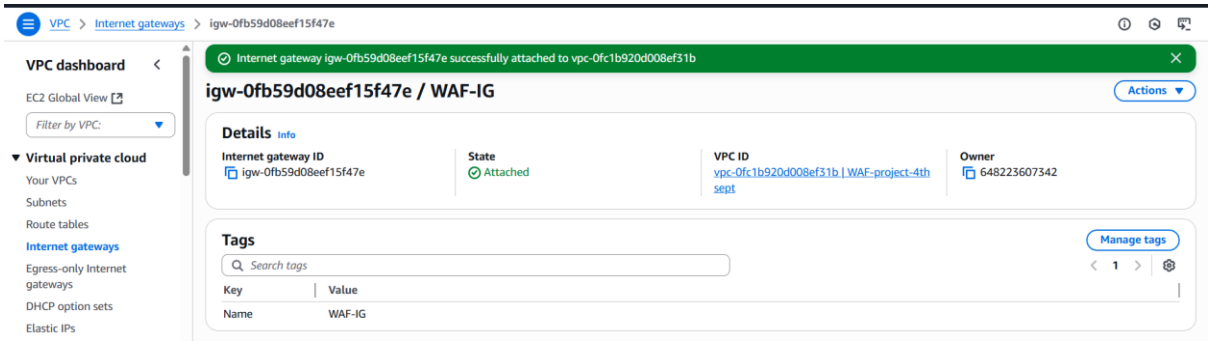
- **Create and Attach Internet Gateway**: Create an Internet Gateway (IGW) and attach it to your VPC. This allows resources in your VPC to communicate with the internet.



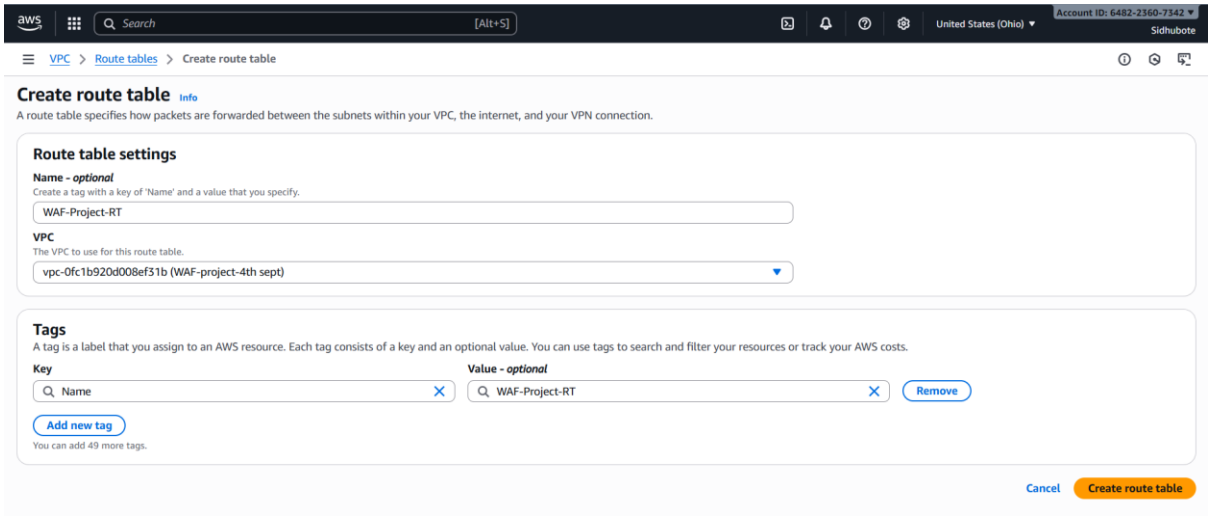
- **Attach VPC**



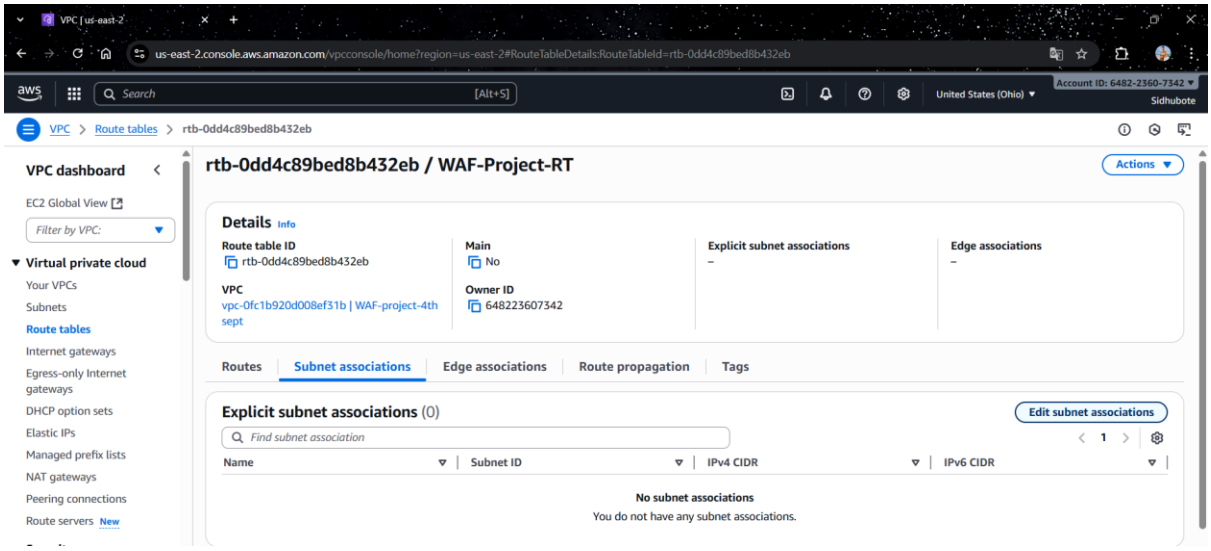
- **Verify VPC**



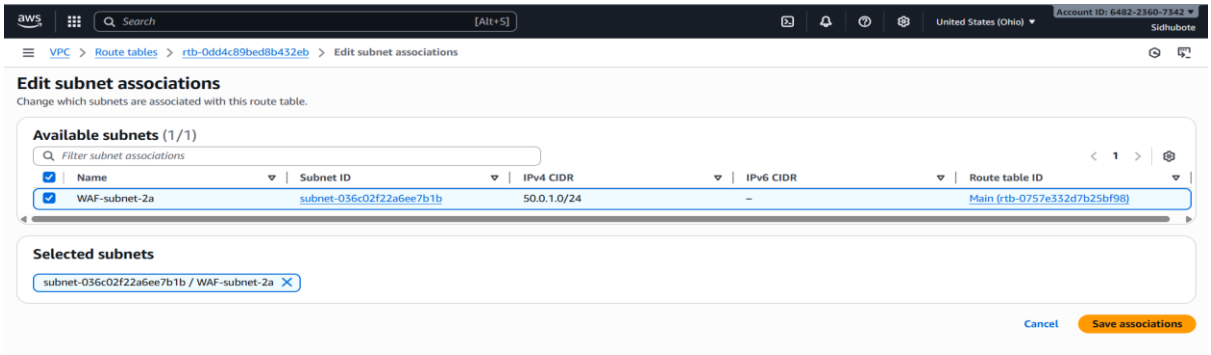
- **Create Route Table**: Create a new route table and associate it with your VPC.



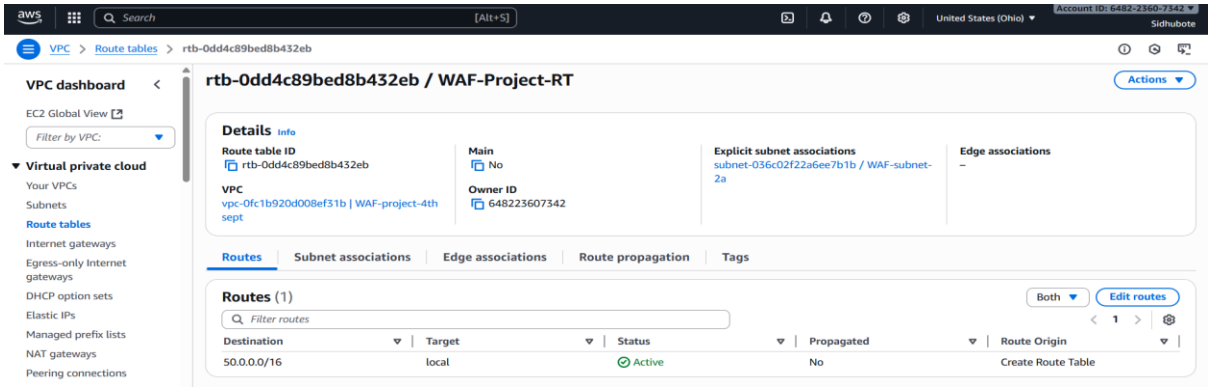
- **Add Route and Subnet Association**: Edit the route table to add a new route with a destination of 0.0.0.0/0 and a target of the Internet Gateway. Then, associate your subnets with this route table to enable them to access the internet.



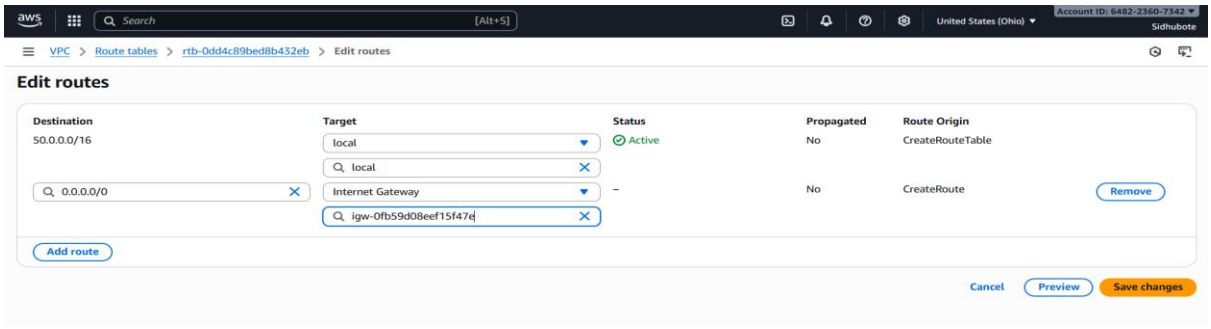
- **Add that subnets**



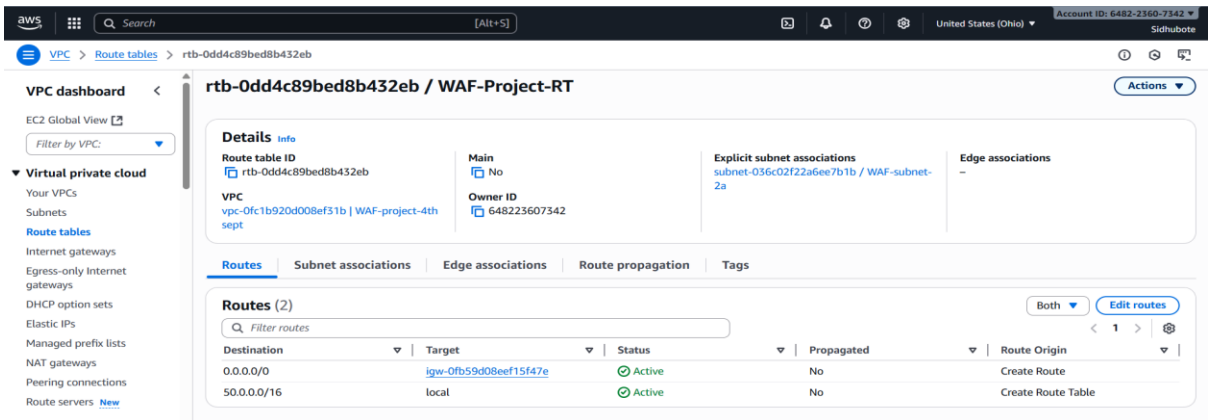
- **Verify subnet Association**



- **Add Route to IG**



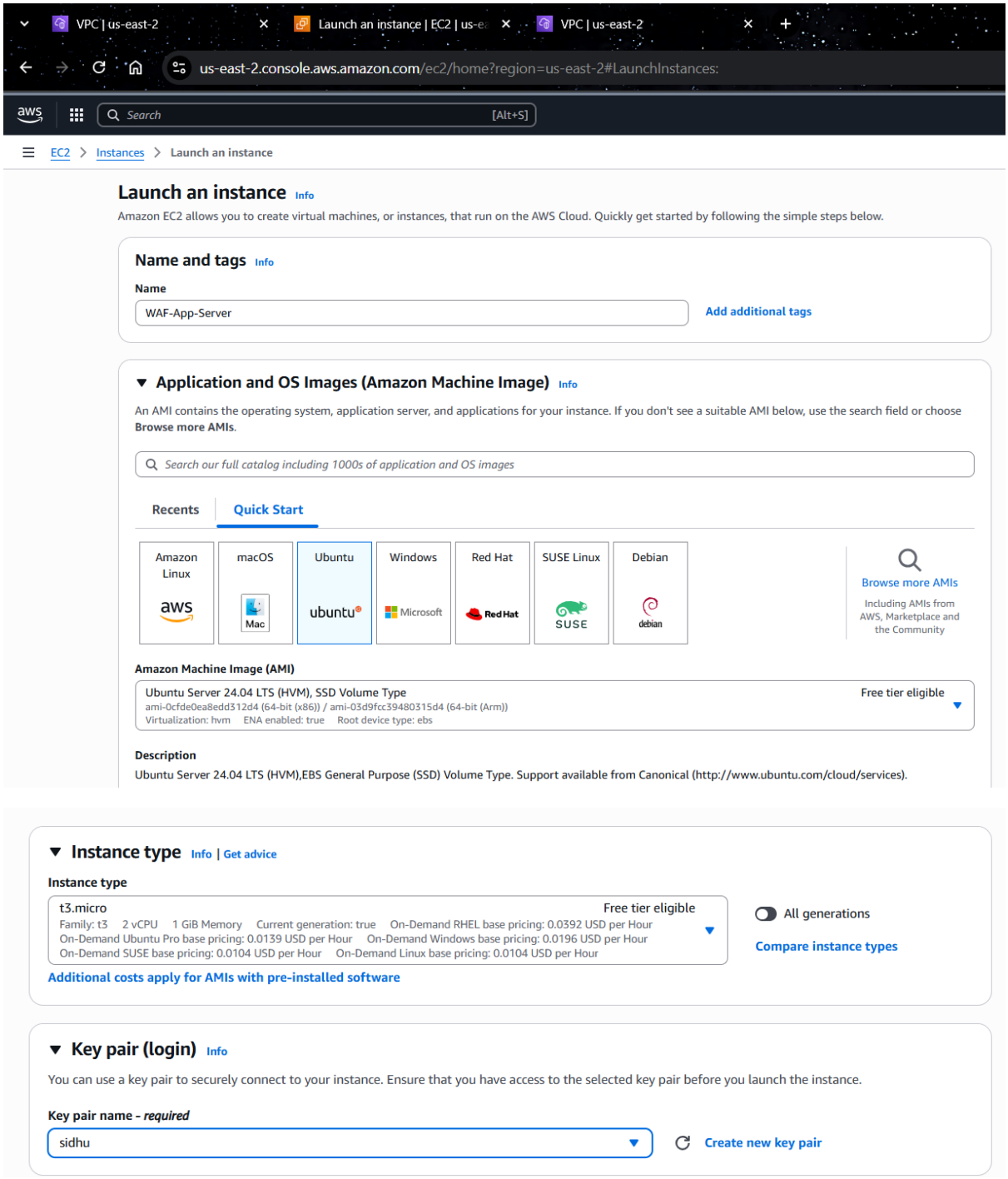
- **Verify Route Table**



2. Launching an EC2 Instance

Next, you will launch an EC2 instance to serve as your web server.

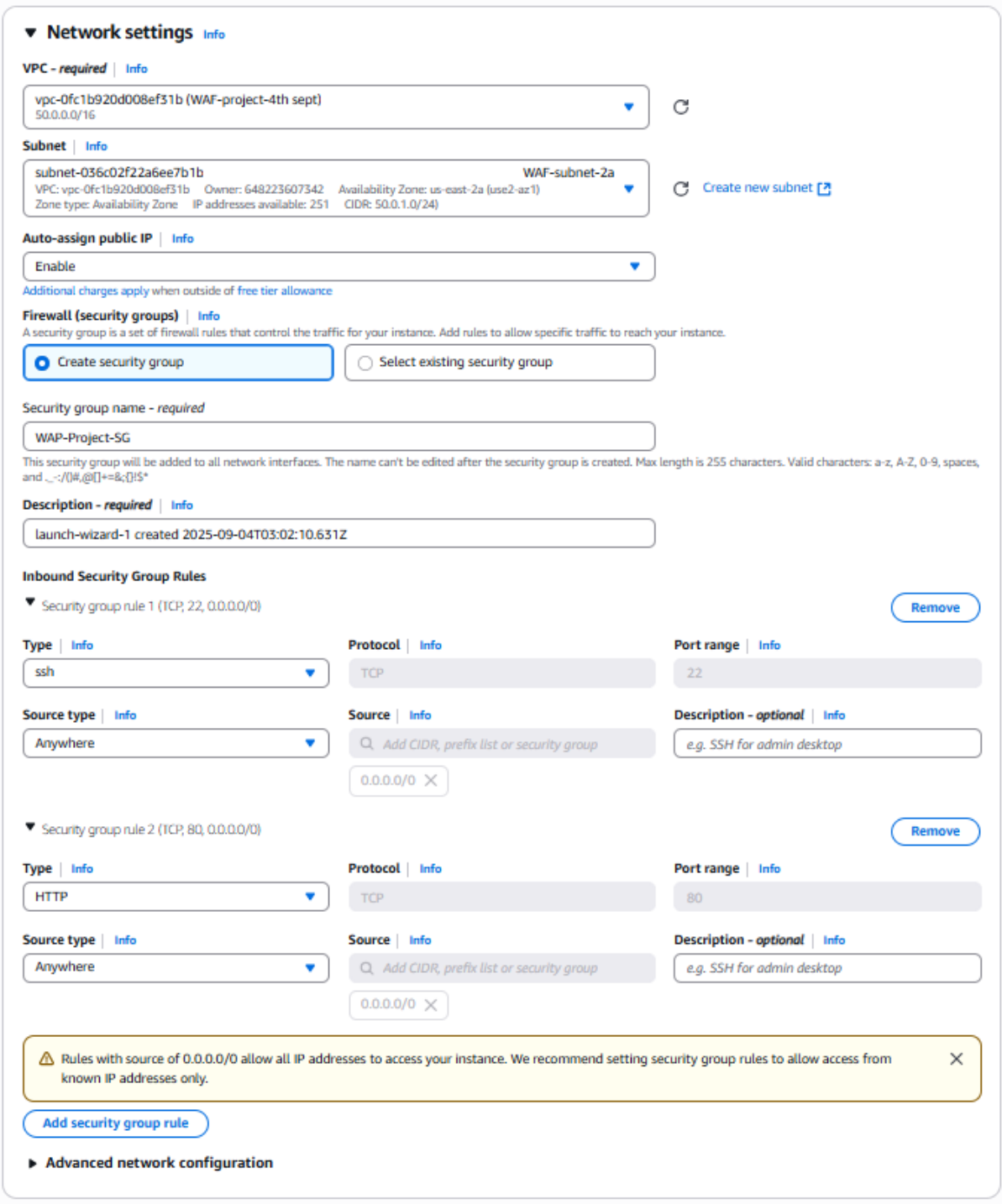
- **Launch Instance**: Go to the EC2 dashboard and select "Launch instance". Name it WAF-App-Server, choose an appropriate AMI (e.g., Ubuntu Server), and an instance type (e.g., t3.micro).



- **Configure Networking**: Select the VPC and one of the subnets you created (WAF-subnet-2a is a good choice for this example). Enable

"Auto-assign public IP" to make the instance accessible from the internet.

- **Create Security Group**: Create a new security group named WAF-Project-SG. Add inbound rules to allow traffic on port 80 (HTTP) and port 22 (SSH) from anywhere (0.0.0.0/0).



- **Add User Data Script**: In the "User data" section, paste the following script. This script will automatically install Apache and host a simple static webpage when the instance launches.

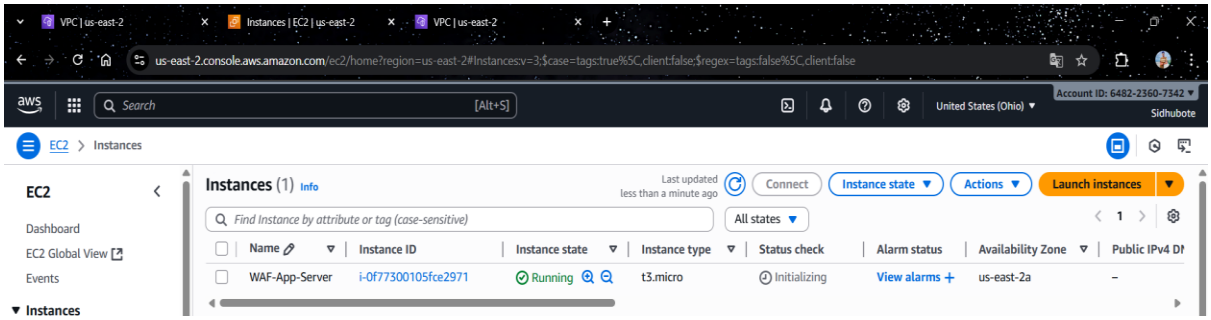
User data - *optional* | Info

Upload a file with your user data or enter it in the field.

Choose file

```
#!/bin/bash
# User Data Script for EC2
# Installs Apache2 and hosts a static website
# Author: Sidhant Bote

# Update system
apt update -y

# Install Apache2
apt install apache2 -y

# Enable and start Apache service
systemctl enable apache2
systemctl start apache2

# Get Hostname and IP
HOSTNAME=$(hostname)
IPADDR=$(hostname -I | awk '{print $1}')

# Create a static index.html
cat > /var/www/html/index.html <<EOF
<!DOCTYPE html>
<html>
<head>
  <title>AWS Web Server</title>
  <style>
    body {
      background-color: #eef2f3;
      font-family: Arial, sans-serif;
      text-align: center;
      padding: 50px;
    }
    h1 { color: #34495e; }
```

- **Verify Instance**: After the instance is running, you can connect via SSH to verify the Apache installation and the webpage content using curl localhost. You can also access the website using the instance's public IP address in a web browser.



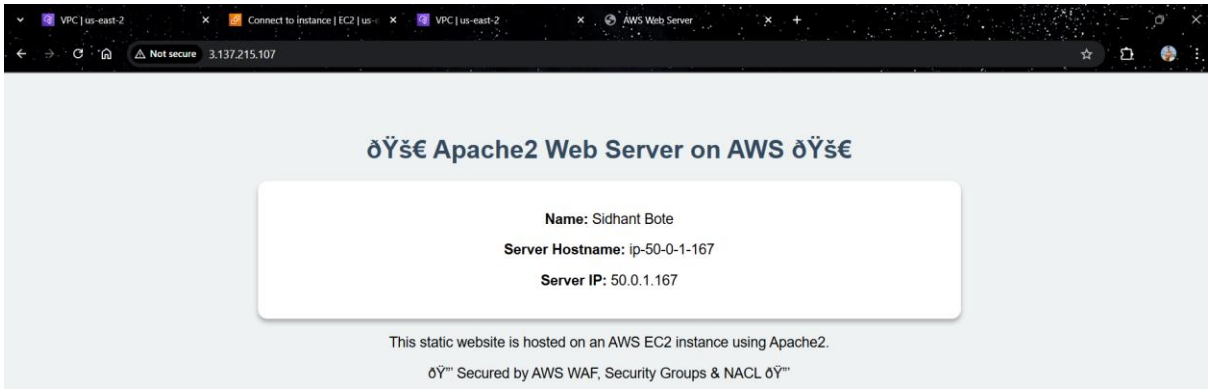
- **Take ssh**

```
PS C:\Users\botes\Downloads> ssh -i "sidhu.pem" ubuntu@3.137.215.107
The authenticity of host '3.137.215.107 (3.137.215.107)' can't be established.
ED25519 key fingerprint is SHA256:/TYVoBojJnPa6E0x3WZ984Y5lpdigM6bNjKGIupNPVQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.137.215.107' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1011-aws x86_64)

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

 System information as of Thu Sep  4 03:07:42 UTC 2025

  System load:  0.3               Temperature:           -273.1 C
  Usage of /:   29.4% of 6.71GB   Processes:             120
  Memory usage: 26%               Users logged in:       0
  Swap usage:   0%                IPv4 address for ens5: 50.0.1.167

Expanded Security Maintenance for Applications is not enabled.

19 updates can be applied immediately.
17 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status



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.

ubuntu@ip-50-0-1-167:~$
```

- **Access application**

```
ubuntu@ip-50-0-1-167:~$ curl localhost
<!DOCTYPE html>
<html>
<head>
    <title>AWS Web Server</title>
    <style>
        body {
            background-color: #eef2f3;
            font-family: Arial, sans-serif;
            text-align: center;
            padding: 50px;
        }
        h1 { color: #34495e; }
        p { font-size: 18px; }
        .card {
            background: #fff;
            padding: 20px;
            margin: 20px auto;
            width: 60%;
            border-radius: 12px;
            box-shadow: 0 4px 6px rgba(0,0,0,0.3);
        }
    </style>
</head>
<body>
    <h1>🚀 Apache2 Web Server on AWS 🚀</h1>
    <div class="card">
        <p><strong>Name:</strong> Sidhant Bote</p>
        <p><strong>Server Hostname:</strong> ip-50-0-1-167</p>
        <p><strong>Server IP:</strong> 50.0.1.167</p>
    </div>
    <p>This static website is hosted on an AWS EC2 instance using Apache2.</p>
    <p>🔒 Secured by AWS WAF, Security Groups & NACL 🔒</p>
</body>
</html>
```

- **Access on browser**

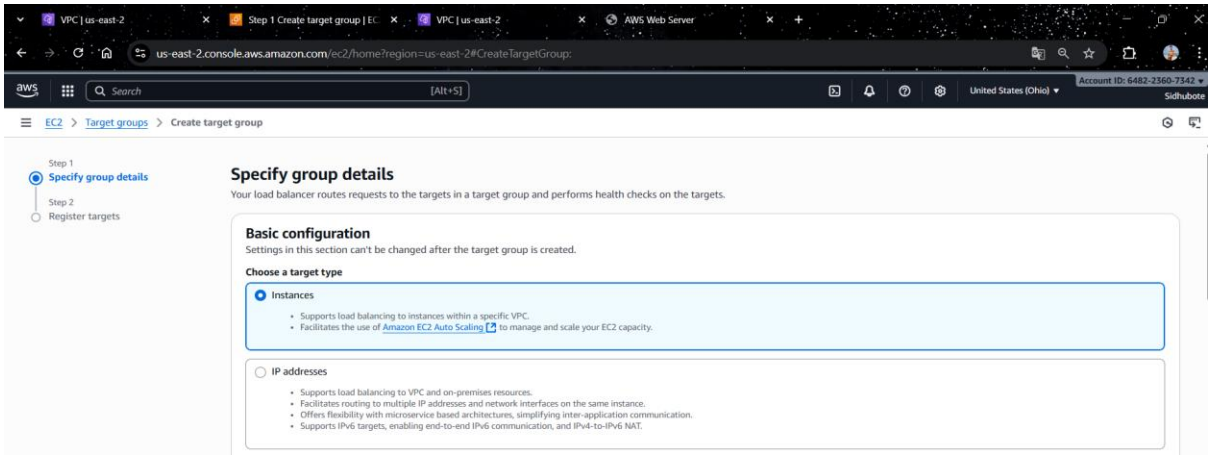


---

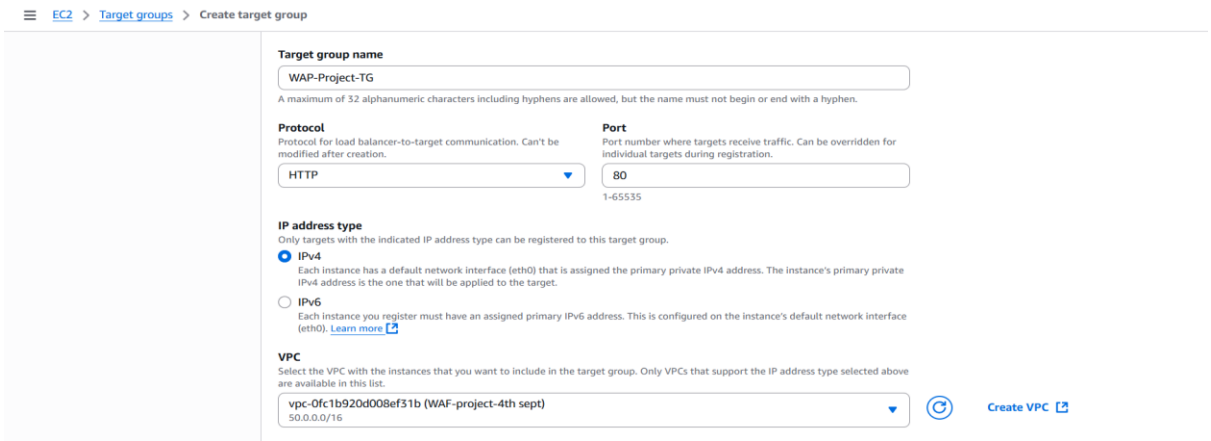
3. Configuring Application Load Balancer

An ALB will distribute traffic to your EC2 instance and will be the resource that WAF protects.

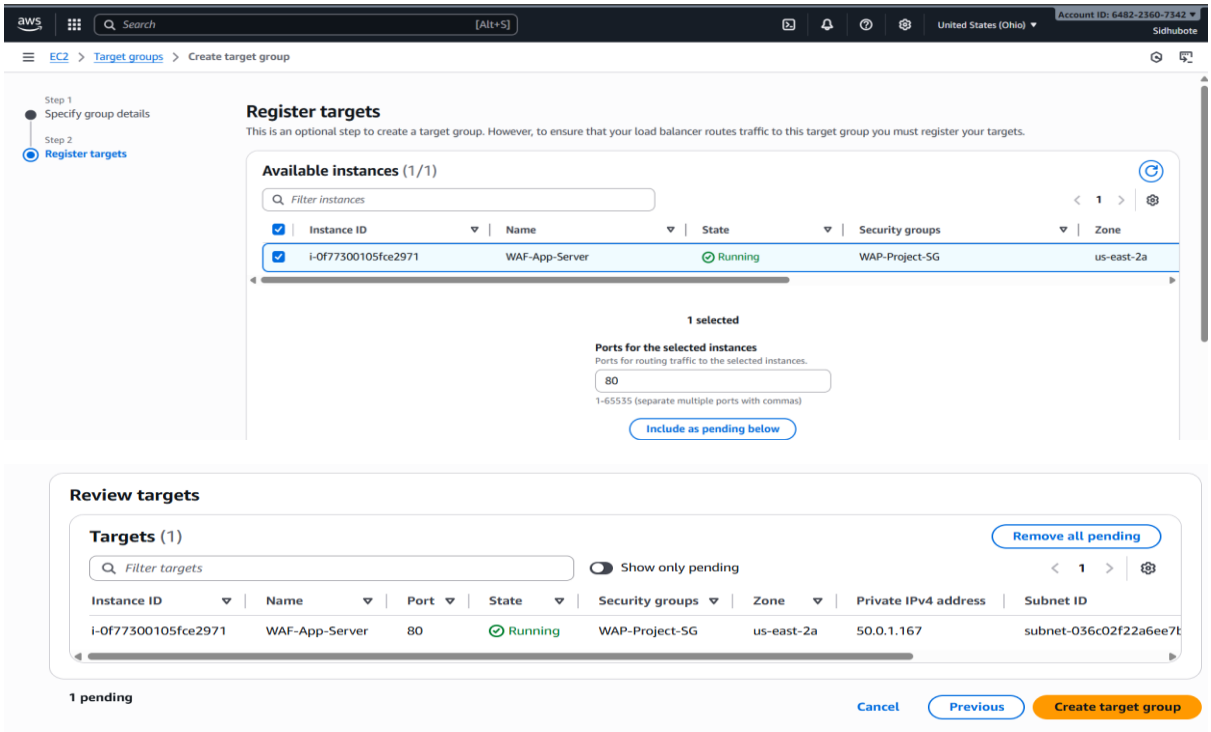
- **Create Target Group**: In the EC2 dashboard under "Load Balancing", create a new Target Group. Choose "Instances" as the target type and name it WAP-Project-TG. Specify the protocol as HTTP on port 80. Add your running WAF-App-Server instance to this target group.



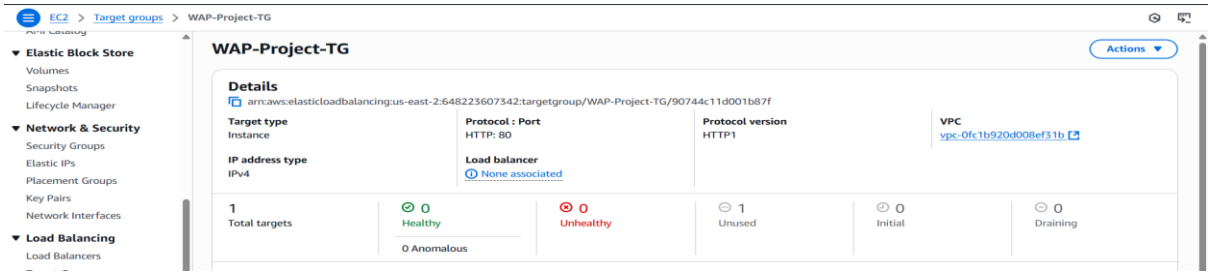
- **Configure the other things.**



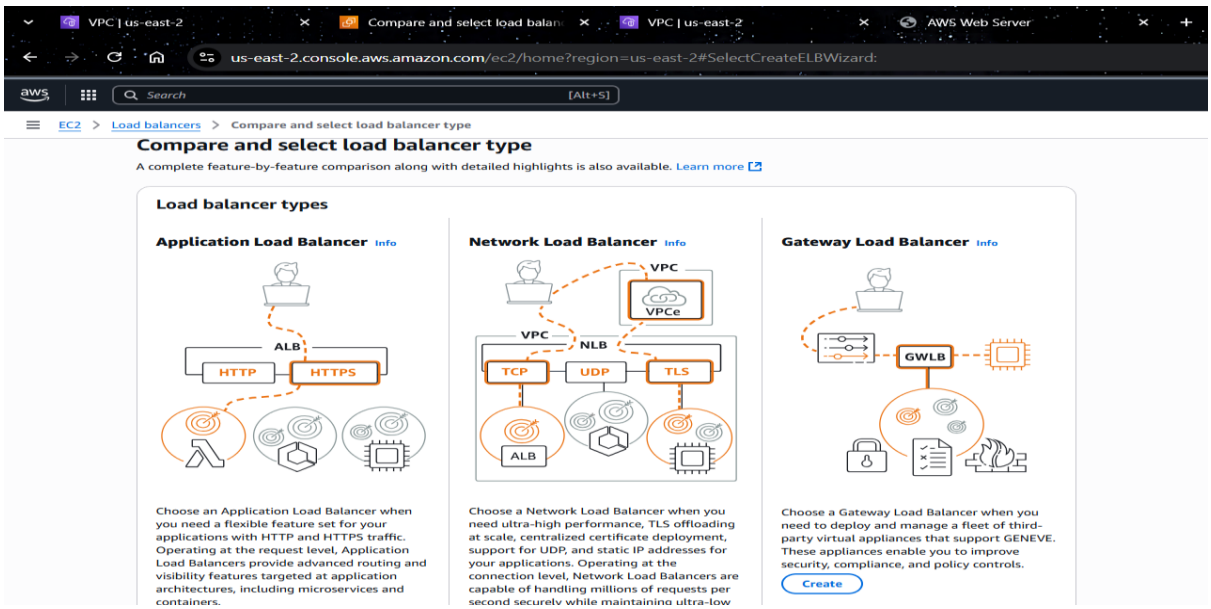
- **Register targets & review target.**



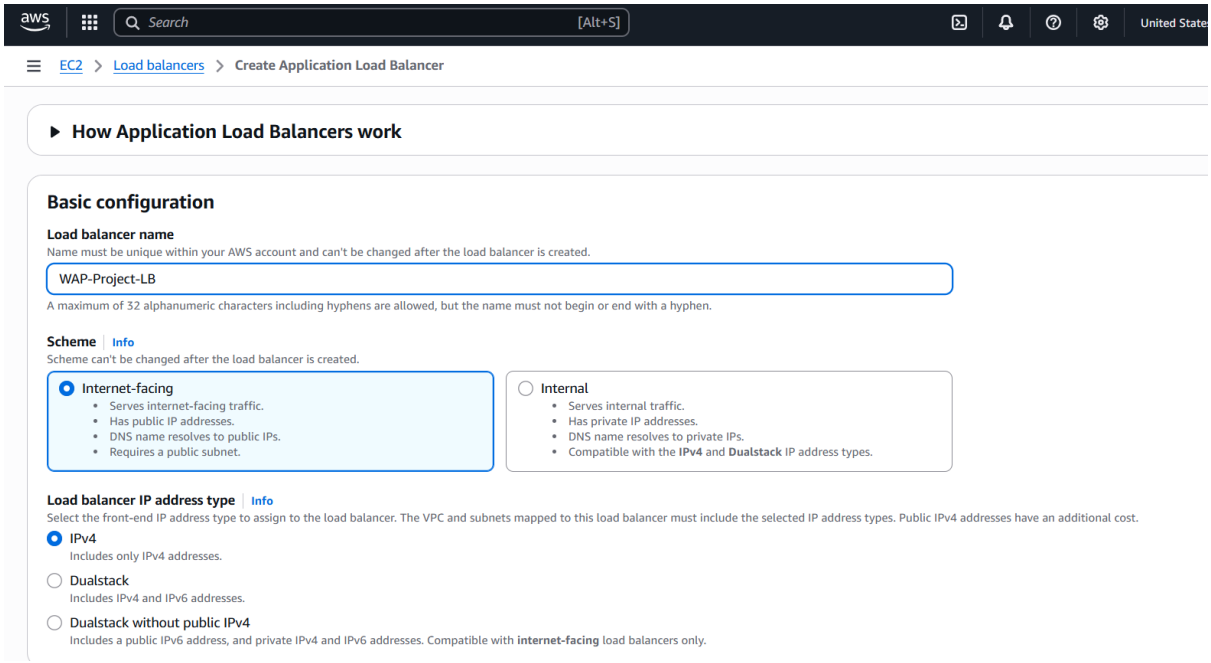
- **Verify Target Group**



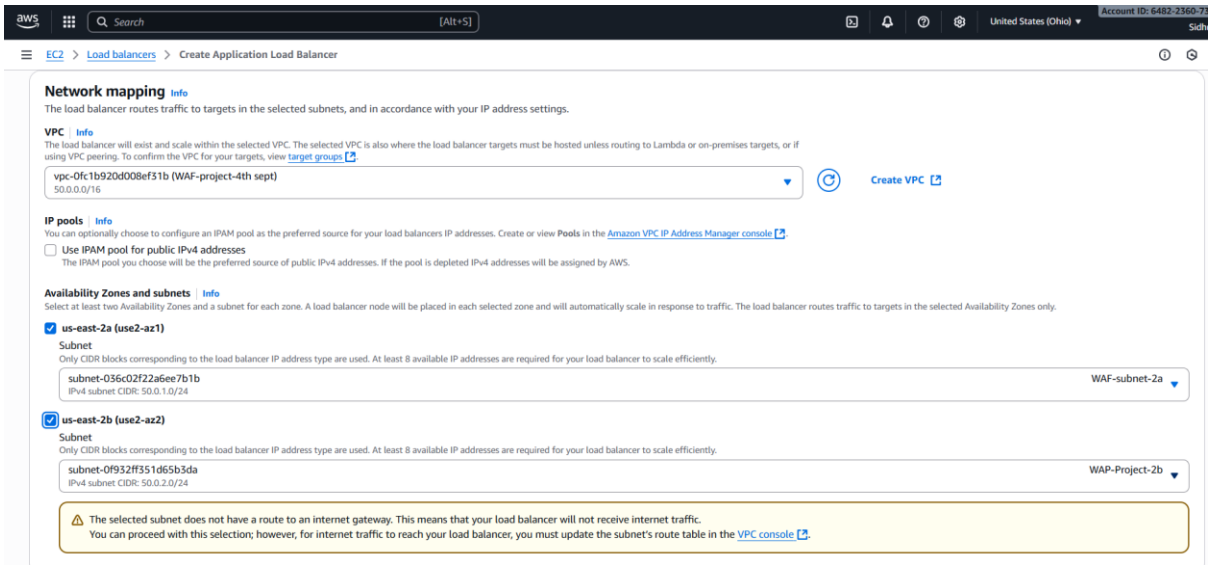
- **Create ALB**: In the "Load Balancers" section, create a new Application Load Balancer.



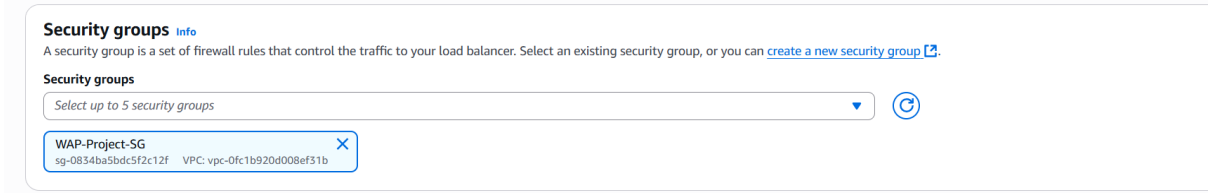
- **Name it WAP-Project-LB and make it "Internet-facing".**



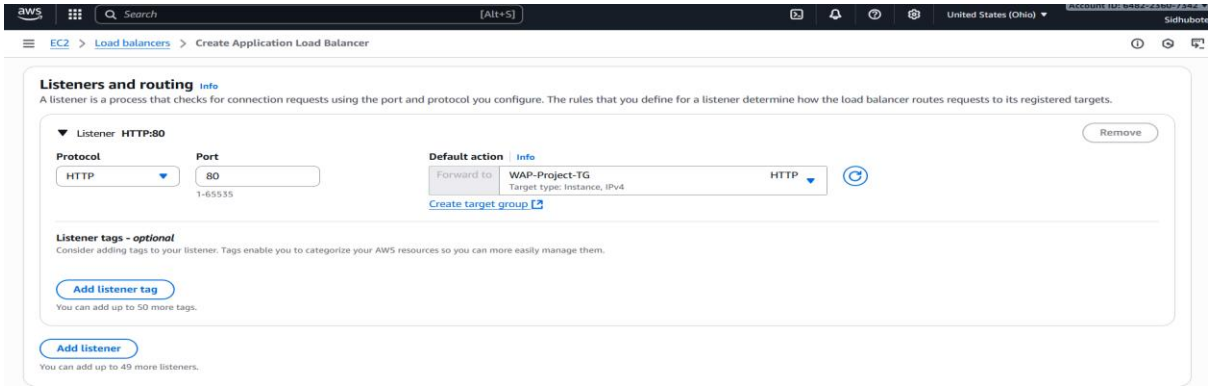
- **Select the VPC you created and both of your subnets.**



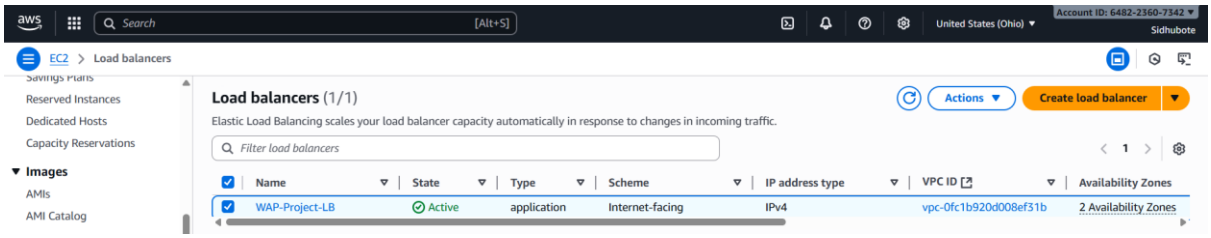
- **Select the same security group as your EC2 instance or a new one that allows inbound traffic on port 80.**



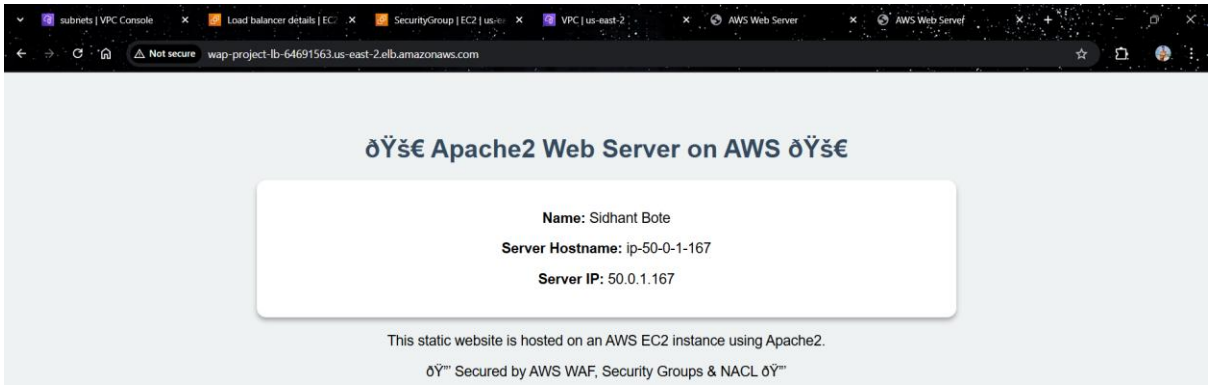
- **Configure the listener for HTTP on port 80 and forward requests to the WAP-Project-TG target group.**



- **Verify ALB**: Once the ALB is active, you can access the website using the ALB's DNS name to verify it is working correctly.



- **Access using DNS in Browser**

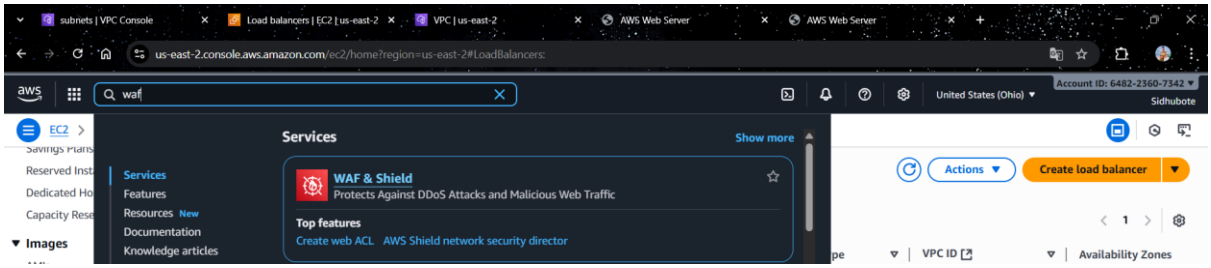


- **Access in the instance CLI**

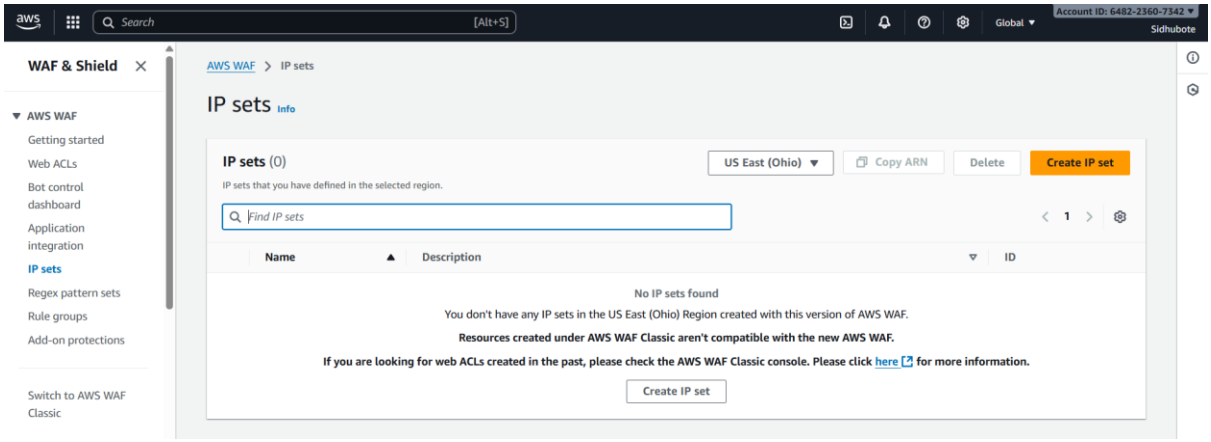
```
ubuntu@ip-50-0-1-167:~$ curl http://wap-project-lb-64691563.us-east-2.elb.amazonaws.com/
<!DOCTYPE html>
<html>
<head>
    <title>AWS Web Server</title>
    <style>
        body {
            background-color: #eef2f3;
            font-family: Arial, sans-serif;
            text-align: center;
            padding: 50px;
        }
        h1 { color: #34495e; }
        p { font-size: 18px; }
        .card {
            background: #fff;
            padding: 20px;
            margin: 20px auto;
            width: 60%;
            border-radius: 12px;
            box-shadow: 0 4px 6px rgba(0,0,0,0.3);
        }
    </style>
</head>
<body>
    <h1>🚀 Apache2 Web Server on AWS 🚀</h1>
    <div class="card">
        <p><strong>Name:</strong> Sidhant Bote</p>
        <p><strong>Server Hostname:</strong> ip-50-0-1-167</p>
        <p><strong>Server IP:</strong> 50.0.1.167</p>
    </div>
    <p>This static website is hosted on an AWS EC2 instance using Apache2.</p>
    <p>🔒 Secured by AWS WAF, Security Groups & NACL 🔒</p>
</body>
</html>
```

4. Configuring AWS WAF

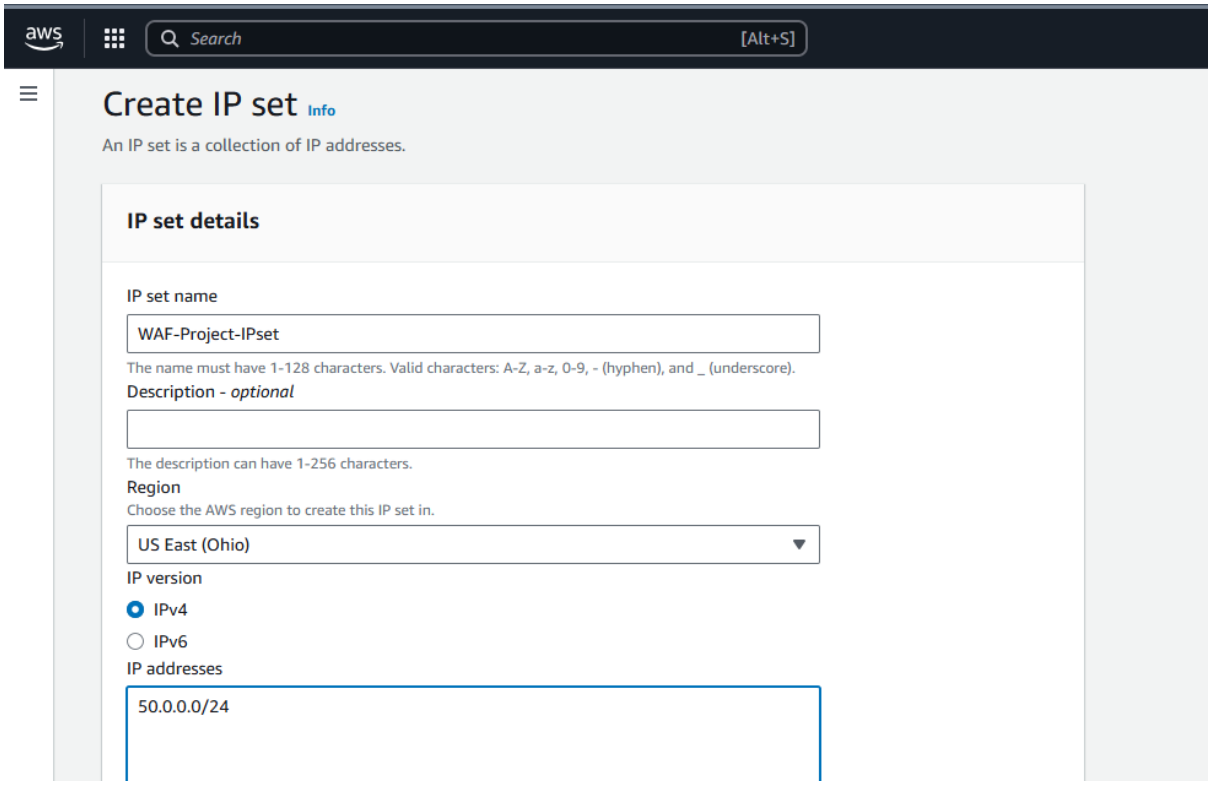
Finally, you will create a Web ACL with a custom rule to protect your application.



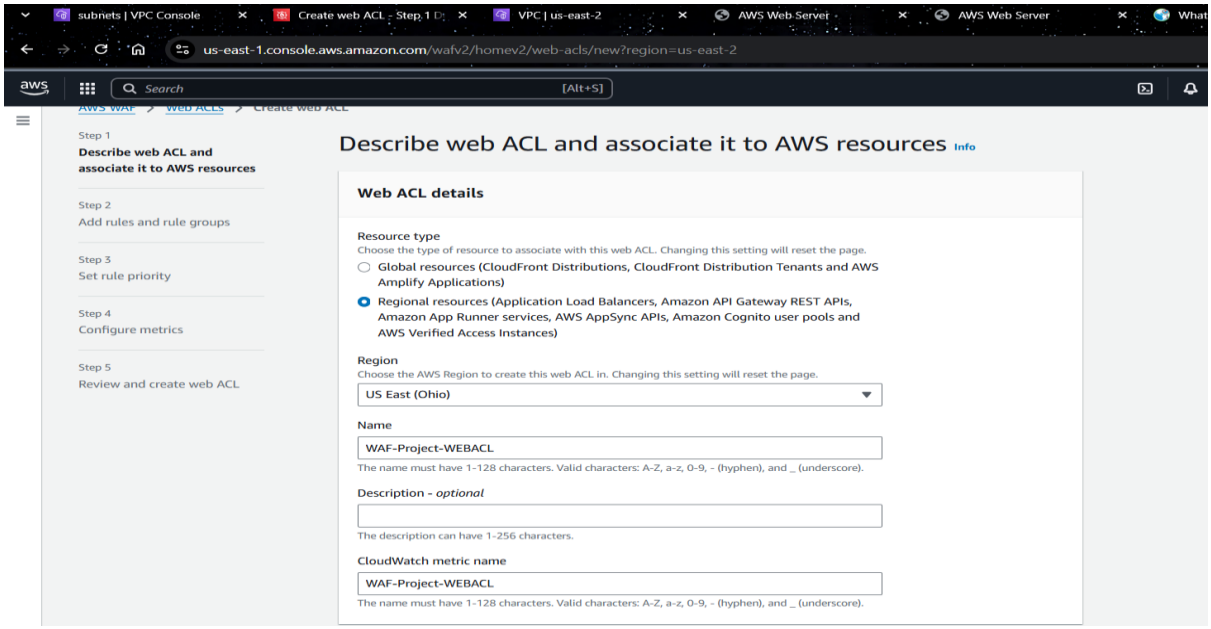
- **Create IP Set**: In the AWS WAF & Shield dashboard, navigate to "IP sets" and create a new IP set.



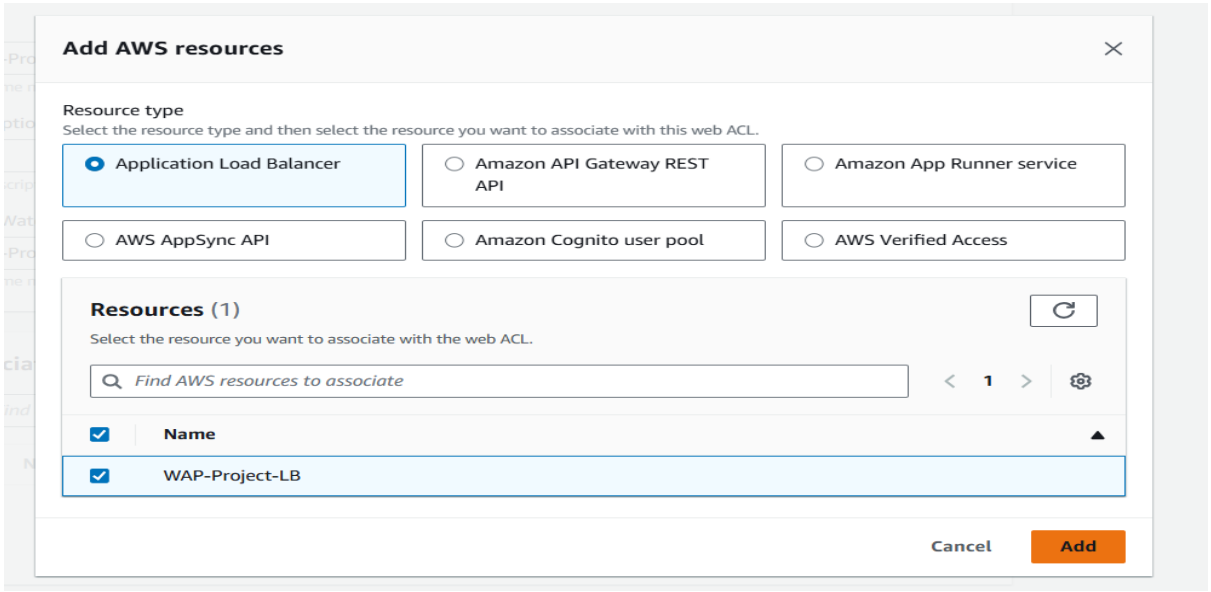
- **Name it WAF-Project-IPset and specify the IP addresses you want to block.**

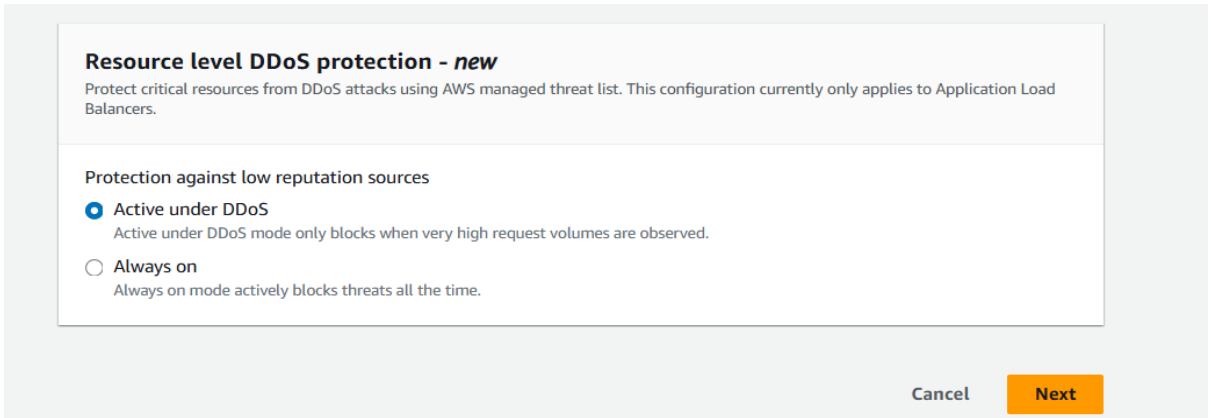


- **Create Web ACL**: Go to "Web ACLs" and create a new Web ACL. Name it WAF-Project-WEBACL and choose "Regional resources".

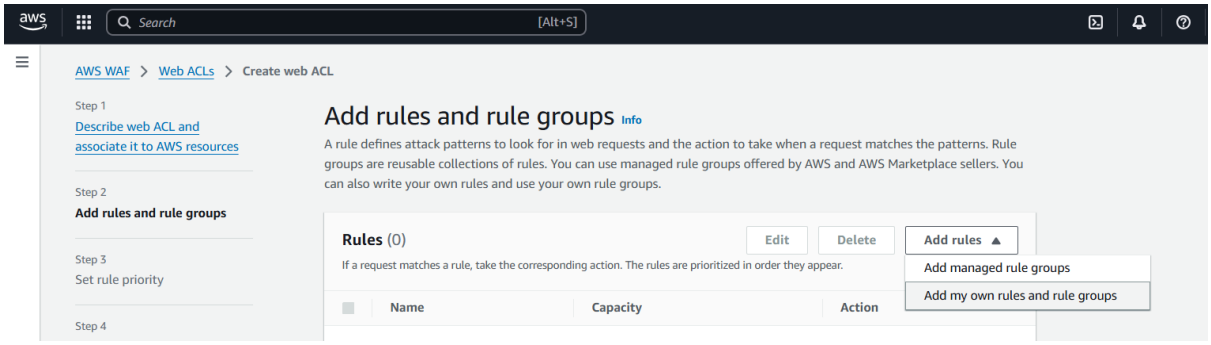


- **Associate the Web ACL with your WAP-Project-LB Application Load Balancer.**

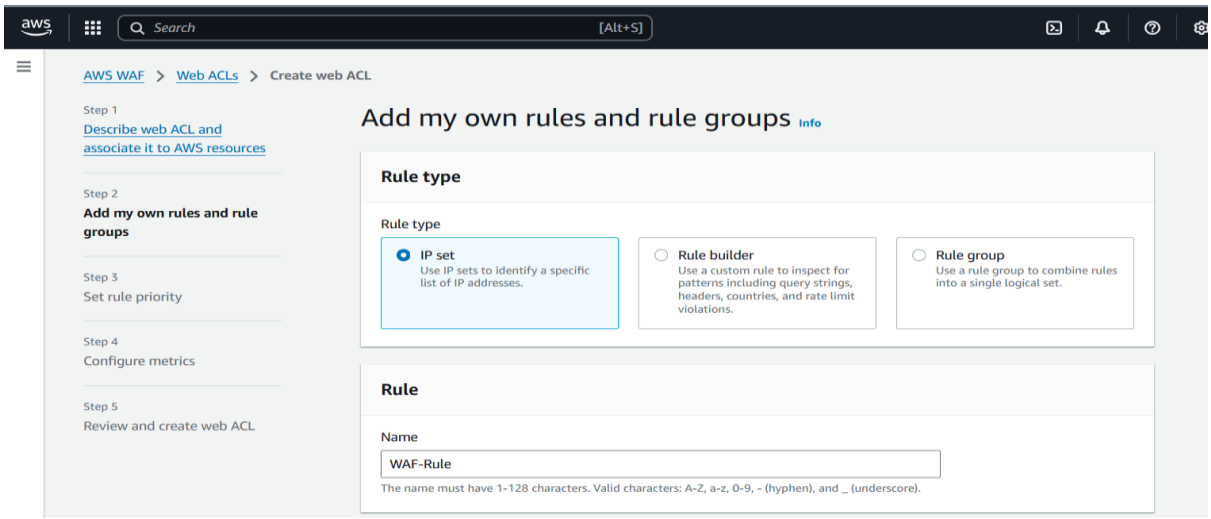




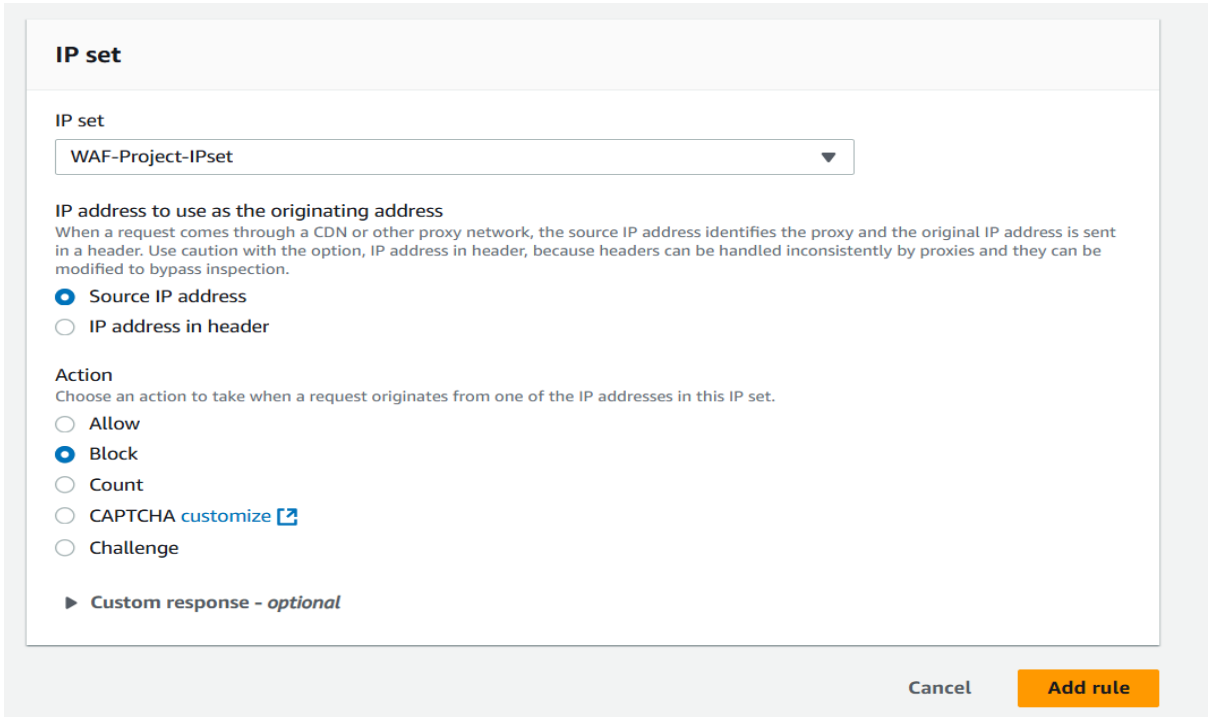
- **Add Custom Rule:** In the "Add rules and rule groups" step, select "Add my own rules and rule groups".



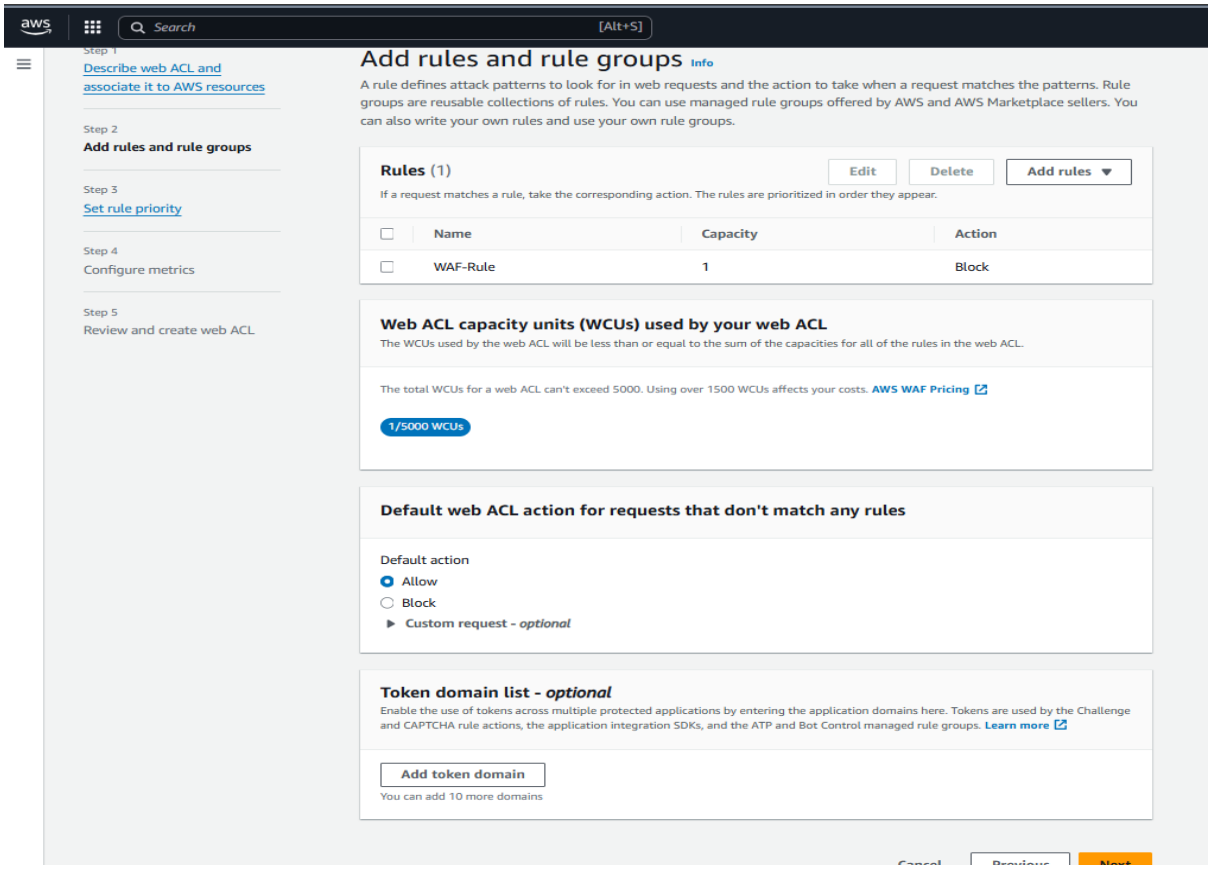
- **Choose "IP set" as the rule type and name it WAF-Rule.**

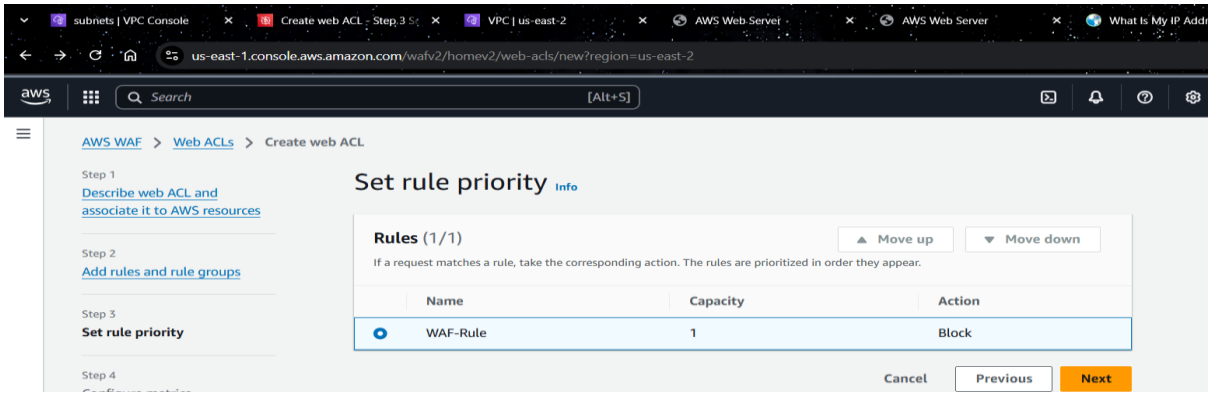


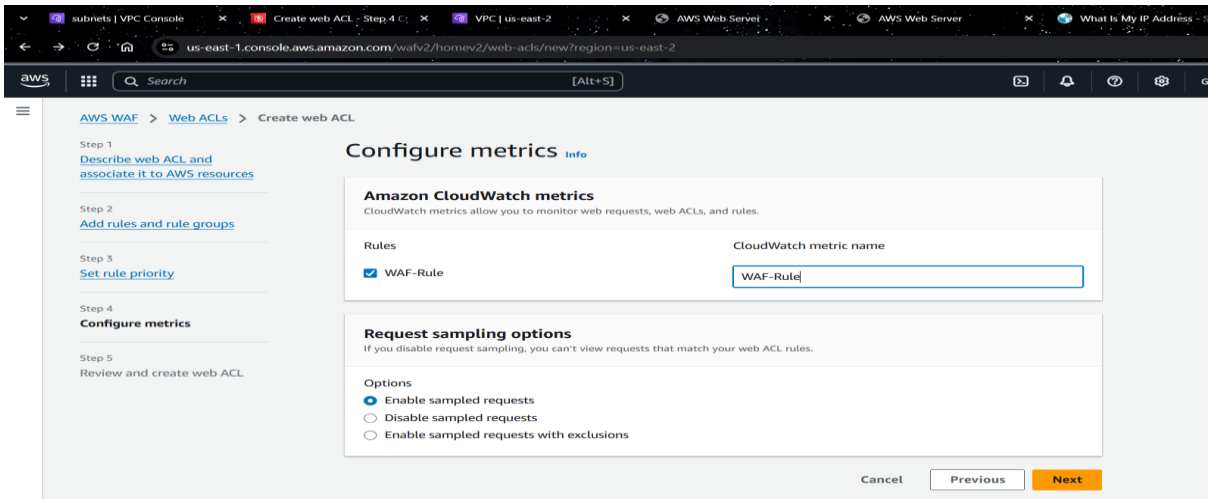
- **Select the WAF-Project-IPset you created and set the action to "Block".**



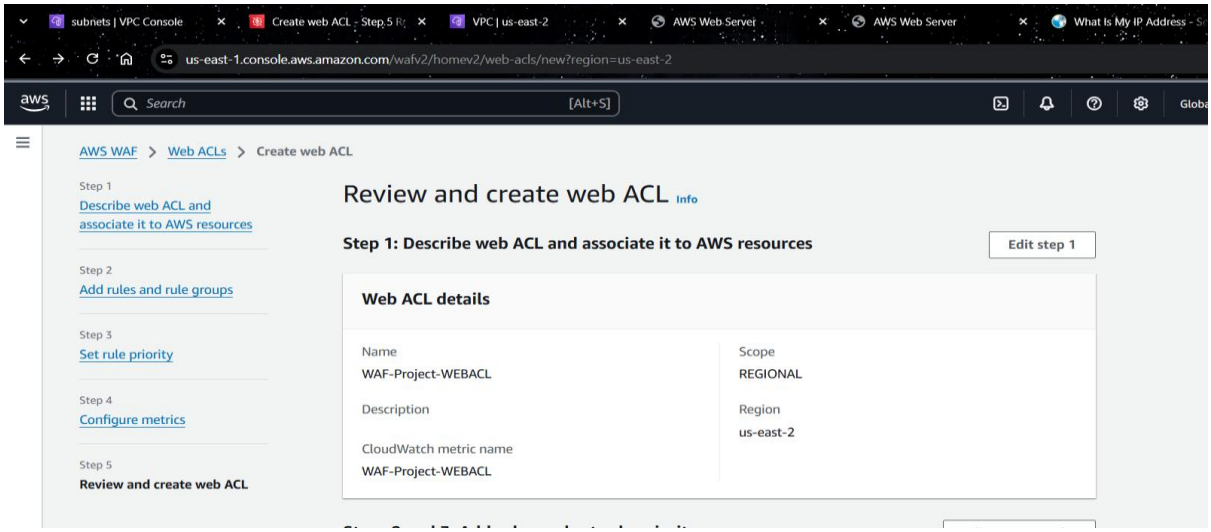
- **Leave Other Configuration Default.**



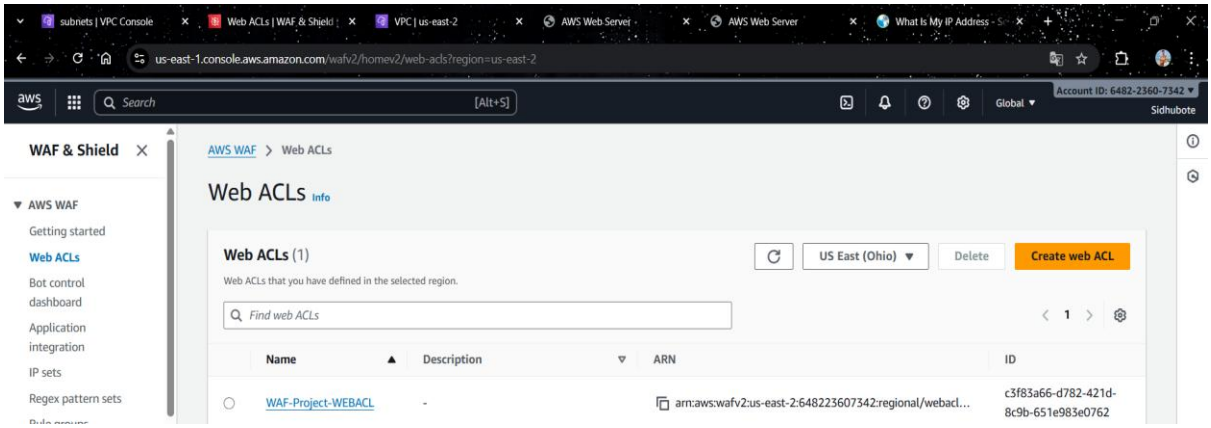




- **Review and Create**: Review the Web ACL settings and create it.



- **Verify WAF**



- **Verify WAF Functionality**: To verify the WAF is working, try to access your ALB's DNS name from an IP address that you have added to the IP set. You should be blocked from accessing the site.

- **Access Using Another IP.**