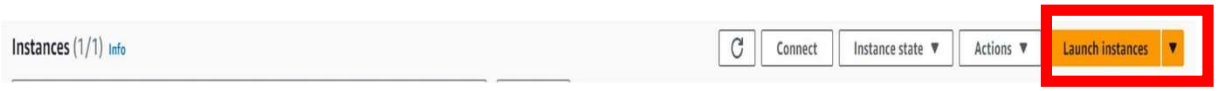
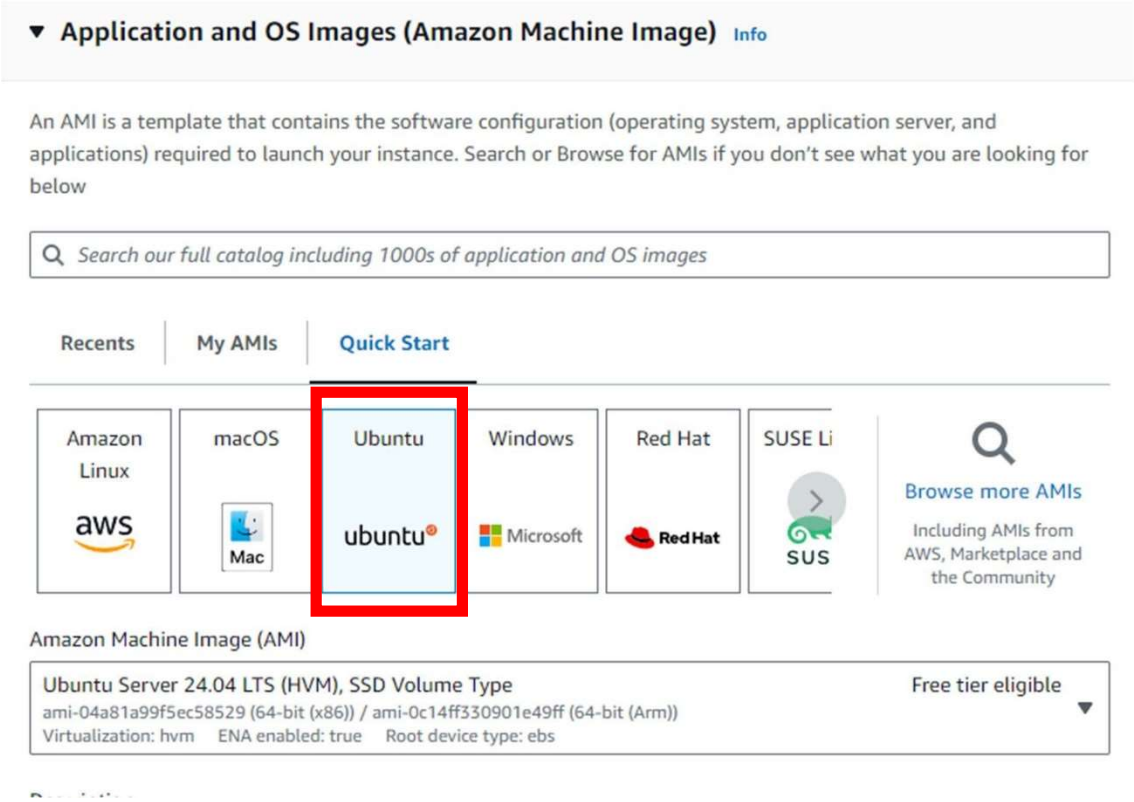
## Step 1: Launch an EC2 Instance

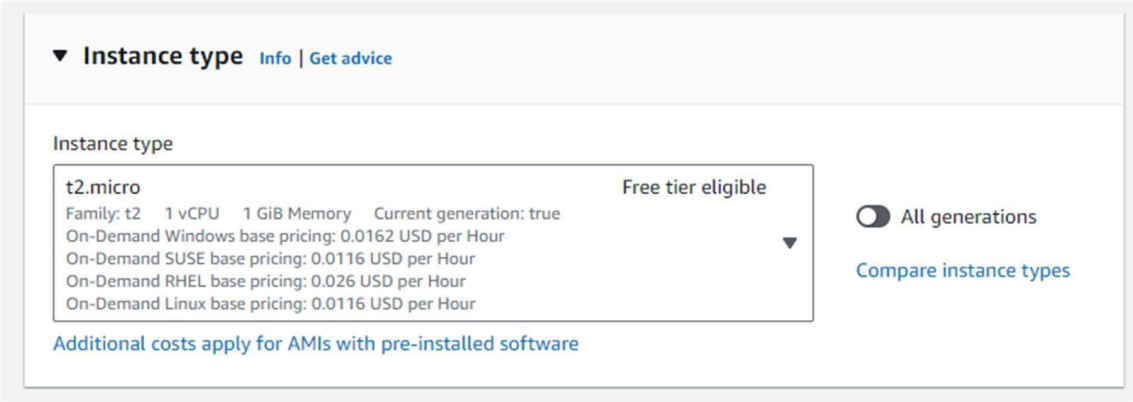
1. **Log in to AWS Management Console**:
   - Go to the AWS Management Console at https://aws.amazon.com/console/
   - Sign in with your AWS credentials.
2. **Navigate to EC2 Dashboard**:
   - In the AWS Management Console, type "EC2" in the search bar and select EC2 to navigate to the EC2 Dashboard.
3. **Launch an Instance**:
   - Click on the "Launch Instance" button.



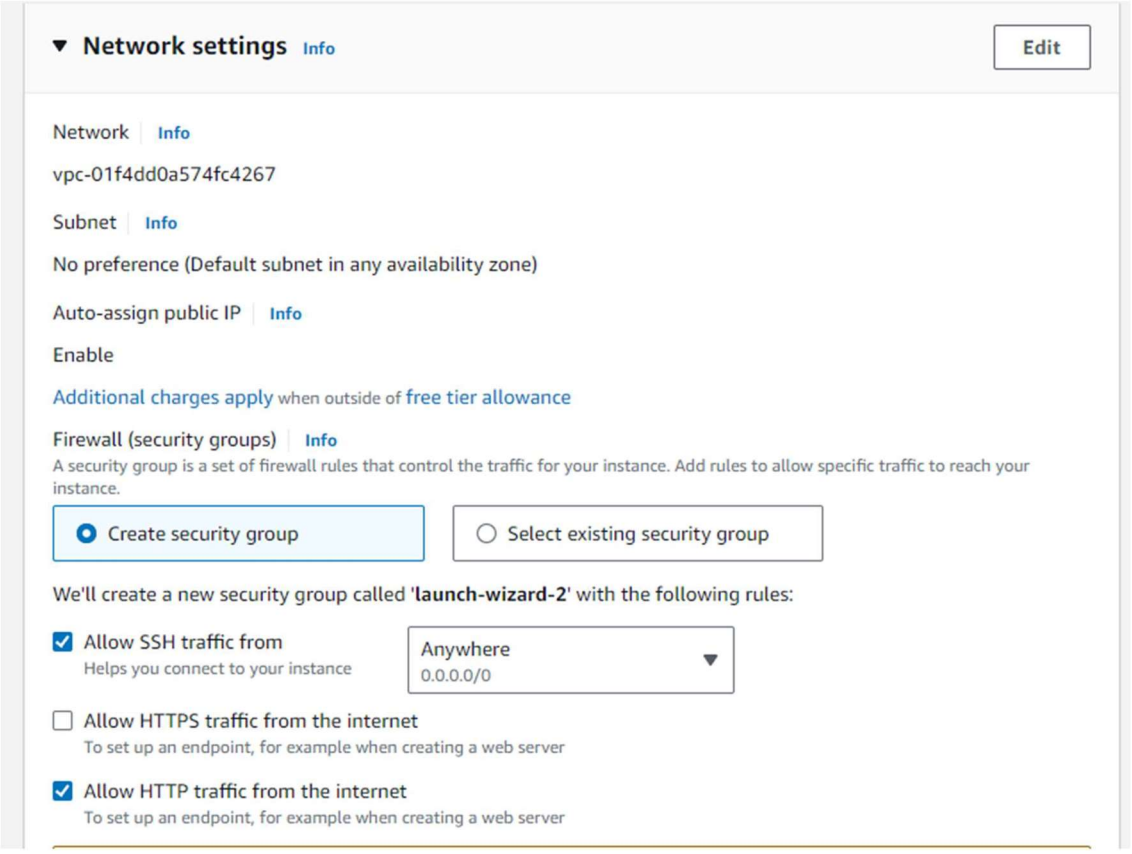

   - Choose an Amazon Machine Image (AMI): Select "Ubuntu Server 20.04 LTS (HVM), SSD Volume Type".




   - Choose an Instance Type: Select `t2.micro` (eligible for the free tier).

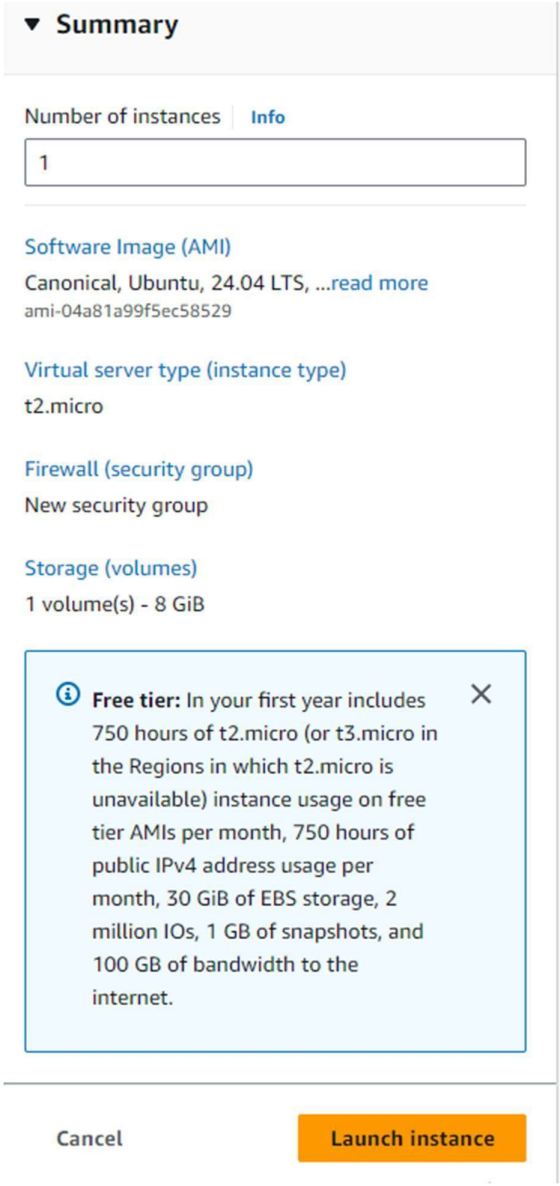


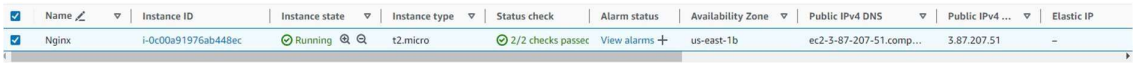

- Configure Instance:
  - Select an existing key pair or create a new one.
  - Network: Choose the default VPC.
  - Subnet: Choose a subnet in the US-East-1 (N. Virginia) region.
  - Enable Auto-assign Public IP.




- Add Storage: Keep the default settings.
- Add Tags: Add a tag to identify your instance (e.g., Key: Name, Value: Nginx).

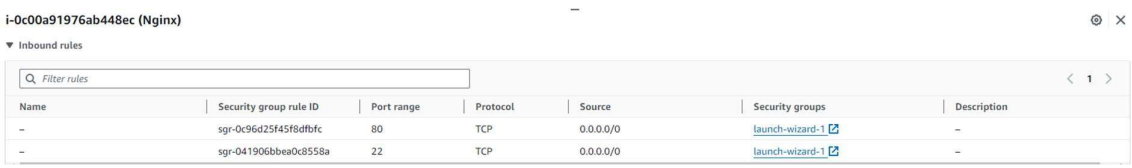
4. **Review and Launch**:
   - Review your instance settings and click "Launch".



| Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability Zone | Public IPv4 DNS | Public IPv4 ... | Elastic IP |
|---|---|---|---|---|---|---|---|---|---|
| Nginx | i-0c00a91976ab448ec | Running | t2.micro | 2/2 checks passed | View alarms + | us-east-1b | ec2-3-87-207-51.comp... | 3.87.207.51 | – |

5. **Configure Security Group:**
   - Add a new security group with the following rules:
     - Type: HTTP, Protocol: TCP, Port Range: 80, Source: 0.0.0.0/0
     - Type: SSH, Protocol: TCP, Port Range: 22, Source: 0.0.0.0/0




## Step 2: Connect to Your Instance

1. **Connect to the EC2 Instance**:
   - In the EC2 Dashboard, select your instance.
   - Click on "Connect" and follow the instructions to connect to your instance using SSH.

## Step 3: Install Nginx

1. **Update Package List**:

```
sudo apt update
```

2. **Install Nginx**:

```
sudo apt install nginx -y
```

3. **Start and Enable Nginx**:

```
sudo systemctl start nginx
sudo systemctl enable nginx
sudo systemctl status nginx
```


## Step 4: Configure Nginx to Display "Hello World"

1. **Modify the Default Nginx Webpage**:
   - Open the default Nginx configuration file:

```
sudo nano /var/www/html/index.nginx-debian.html
```

- Replace the content with the following HTML:

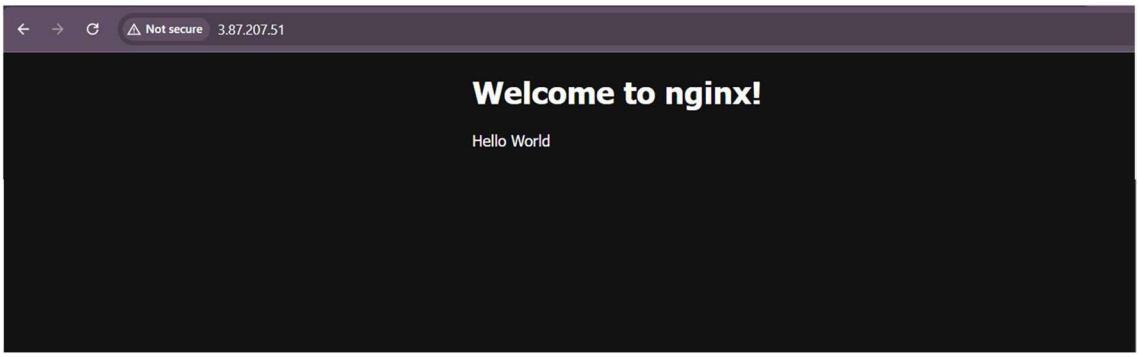
```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to Nginx!</title>
</head>
<body>
    <h1>Hello World</h1>
</body>
</html>
```

```
  GNU nano 7.2
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Hello World</P>
</body>
</html>
```

2. **Save and Close the File**:
   - Press `Ctrl + X` to close the file.
   - Press `Y` to confirm changes, then press `Enter`.

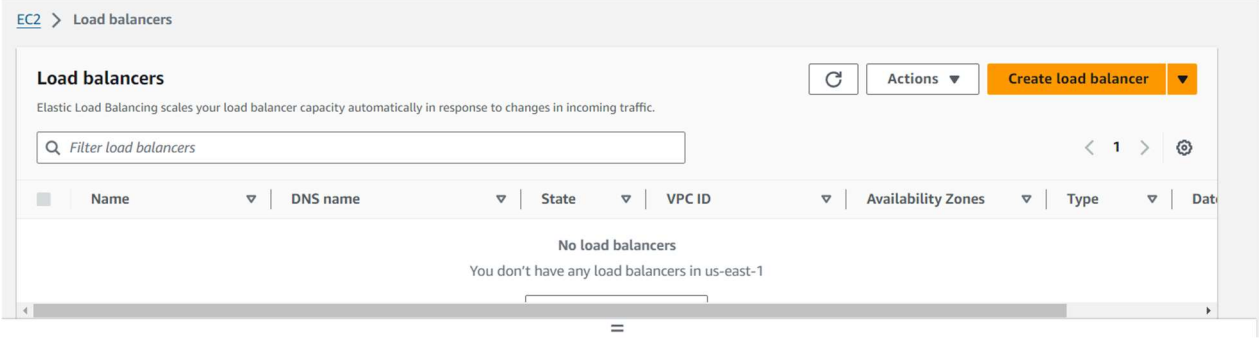
## Step 5: Verify the Configuration

1. **Open a Web Browser**:
   - Enter the public IP address of your EC2 instance in the address bar.
   - You should see a webpage displaying the message: "Hello World".

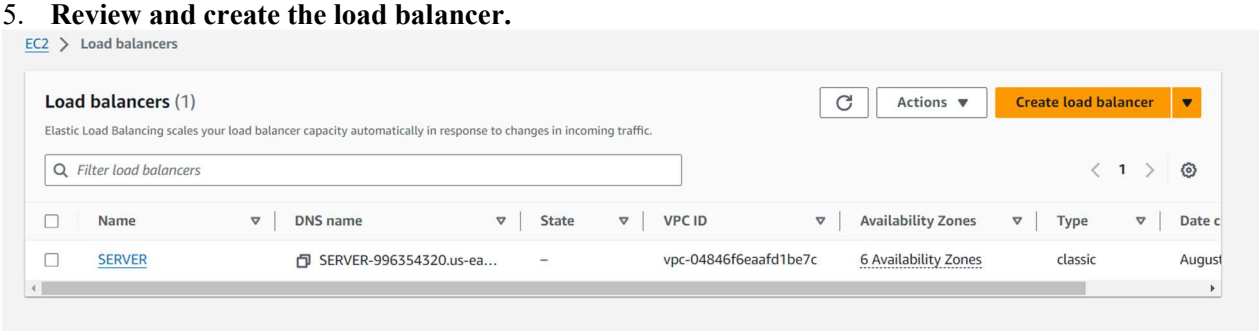



## Step 6: Create a Classic Load Balancer

1. **Go to the EC2 Dashboard.**
2. In the left navigation pane, under **Load Balancing**, select **Load Balancers**.

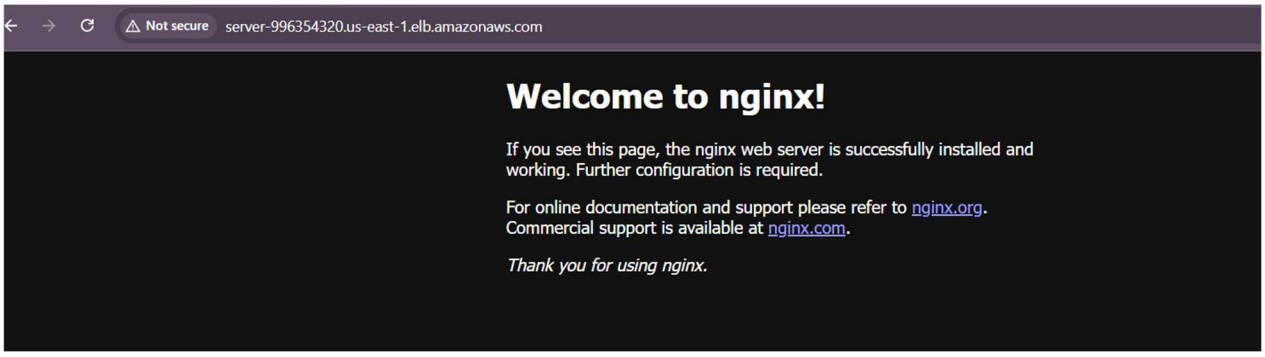



3. Click on **Create Load Balancer** and select **Classic Load Balancer**.
4. **Configure the Load Balancer**:
   - Name your load balancer.
   - Select the VPC and availability zones.
   - Configure listeners (e.g., HTTP on port 80).
   - Configure health checks (e.g., HTTP on /).
   - Select the instances you launched earlier to register with the load balancer.
5. **Review and create the load balancer.**




**Step 4: Test the Classic Load Balancer**

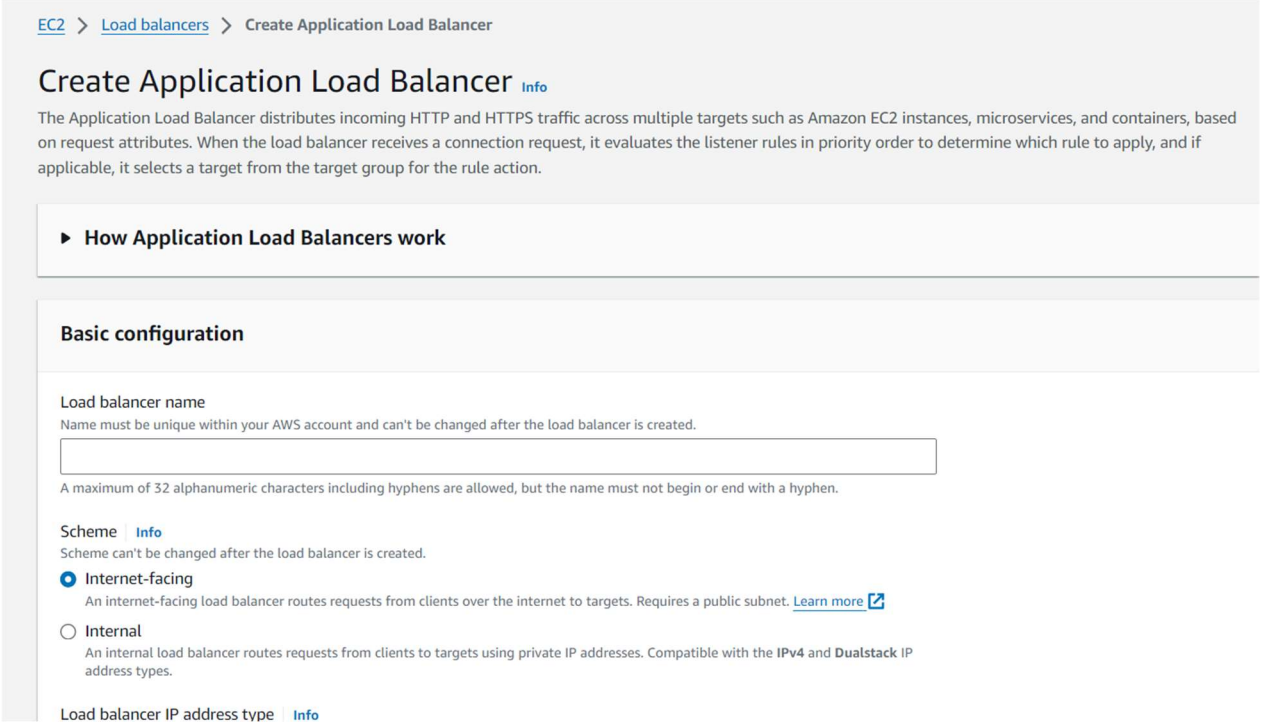
1. **Get the DNS name of the CLB** from the Load Balancers dashboard.
2. Access the DNS name in your web browser to see the load balancing in action.




**Part 2: Migrate to an Application Load Balancer**

**Step 1: Create an Application Load Balancer**

1. **Go to the EC2 Dashboard.**
2. In the left navigation pane, under **Load Balancing**, select **Load Balancers**.
3. Click on **Create Load Balancer** and select **Application Load Balancer**.




**Step 2: Create a Load Balancer**

1. **Navigate to the EC2 Dashboard**:
    - Click on **Load Balancers** under the Load Balancing section.
    - Click on **Create Load Balancer**.
    - Choose **Application Load Balancer**.
    - Configure the load balancer:
        - Name: my-load-balancer.
        - Scheme: Internet-facing.
        - Listeners: HTTP (port 80).
        - Availability Zones: Select the VPC and subnets.
2. **Configure Security Groups** for the load balancer:
    - Ensure it allows HTTP traffic.
3. **Configure Routing**:
    - Create a target group:
        - Name: my-target-group.
        - Target type: Instances.
        - Protocol: HTTP.
        - Port: 80.
        - Health checks: HTTP.

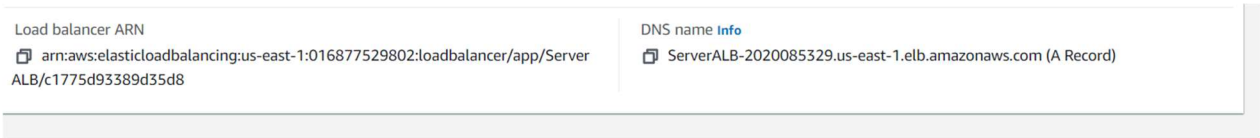
- Register your instances in the target group.

4. **Configure Routing**:
   - Create a new target group.
   - Select Instance as the target type.
   - Register the same EC2 instances you used with the CLB.

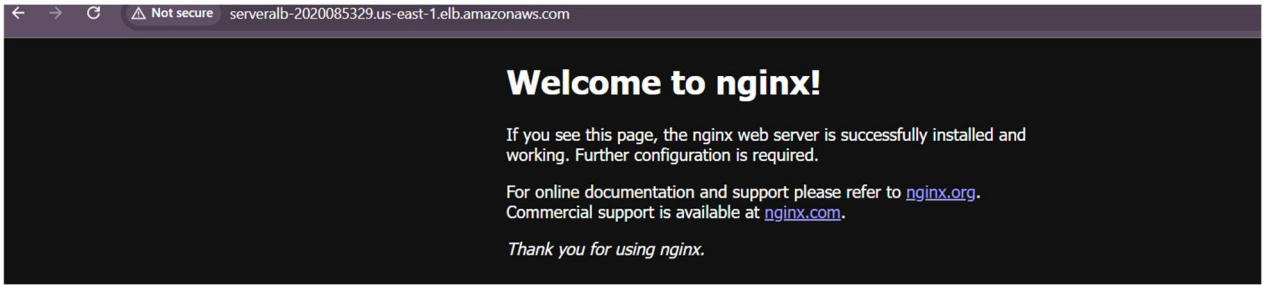
5. **Review and Create** the load balancer.

**Step 4: Test the Application Load Balancer**

1. **Get the DNS name of the ALB** from the Load Balancers dashboard.




2. Access the DNS name in your web browser to ensure the ALB is routing traffic correctly to your instances.




**Step 5: (Optional) Delete the Classic Load Balancer**

- Once you have confirmed the ALB is working correctly, you can delete the Classic Load Balancer if it is no longer needed.

**Note**

- Ensure that your security groups allow inbound traffic on port 80 for the ALB and instances.

This process will set up a Classic Load Balancer with registered EC2 instances and then migrate to an Application Load Balancer while ensuring the web pages remain accessible.