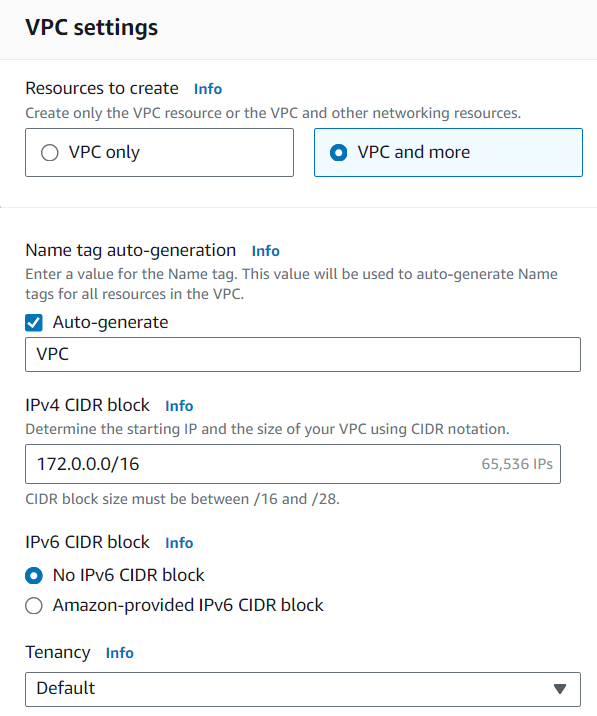
**PROJECT-1**

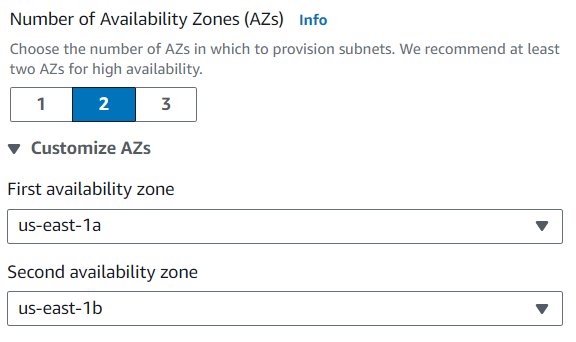
**3-TIER ARCHITECTURE**

****

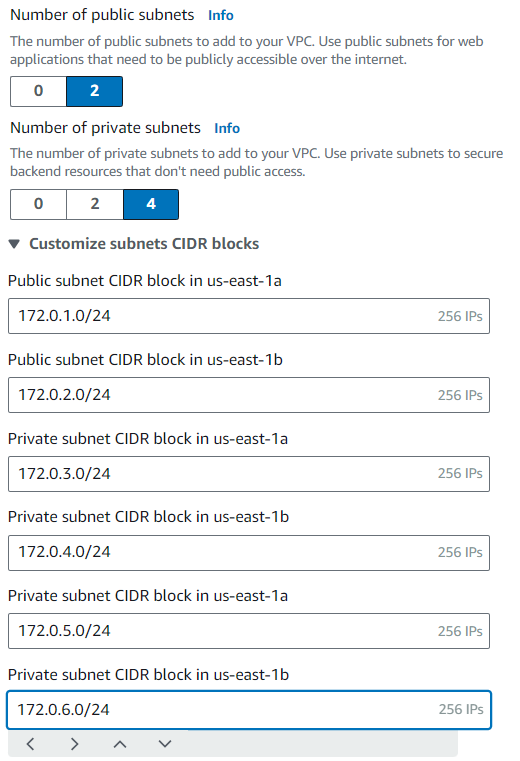
Go to **VPC > Create VPC > Select VPC and more option.** Then, keep the IPv4 CIDR block as 10.0.0.0/16 and click **No** for IPv6 CIDR block. Keep **Tenancy** as **Default**.



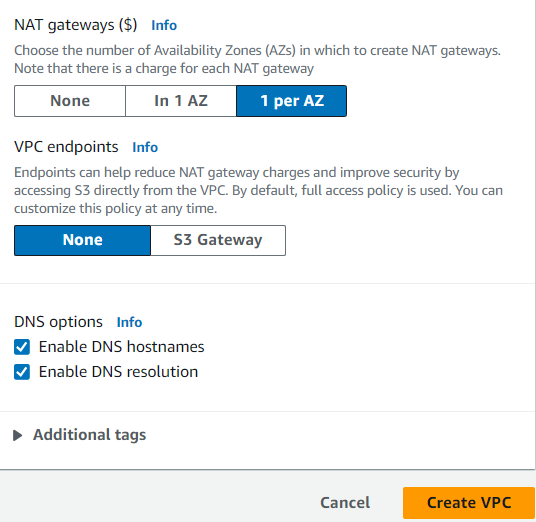
Select ‘2’ **Availability Zone**and select the **First availability zone** as us-east-1a and the **Second availability zone** as us-east-1b.



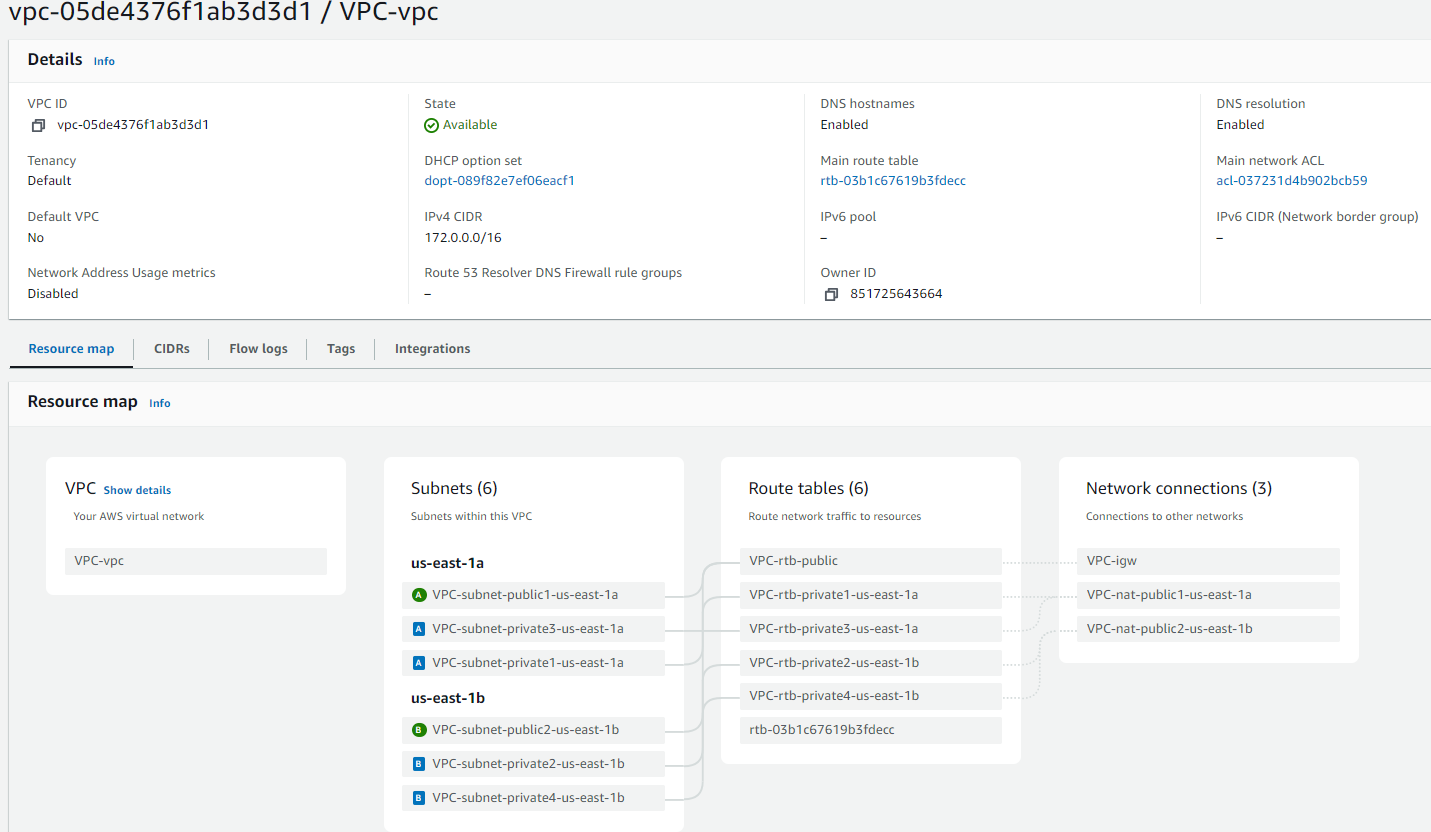
Enter the **public and private subnet CIDR**information as shown below.



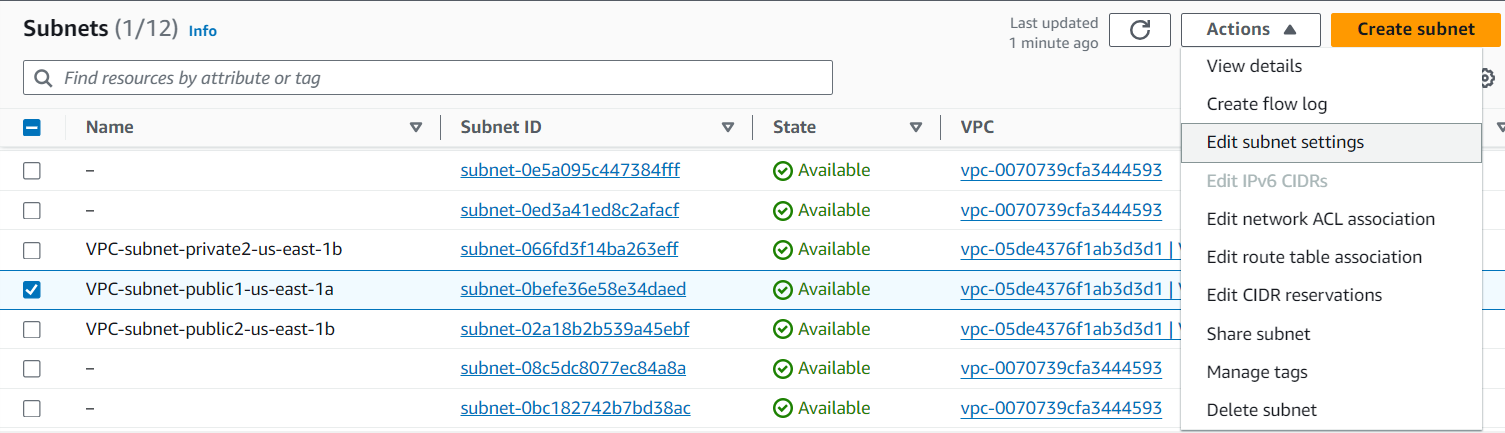
Choose **1 per AZ** for **NAT gateway** and **None** for **VPC endpoints**. Enable **DNS hostnames**and **DNS resolution**. Click **Create VPC** button.

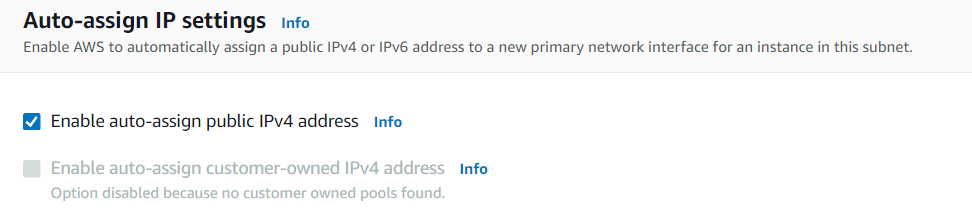


Once completed, click **View VPC** button. Now, we have a VPC, 2 public subnets, 4 private subnets, and 2 NAT gateways.

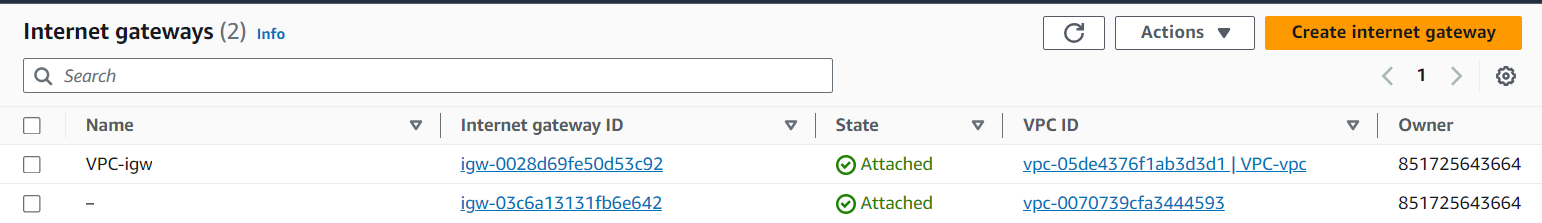


Go to **VPC > Subnets**>Select one of two public subnets and then click **Actions**menu button. Click **Edit subnet settings**. Enable **auto-assign public IPV4 address**. Click **Save** button. Repeat step 6 for the other public subnet.

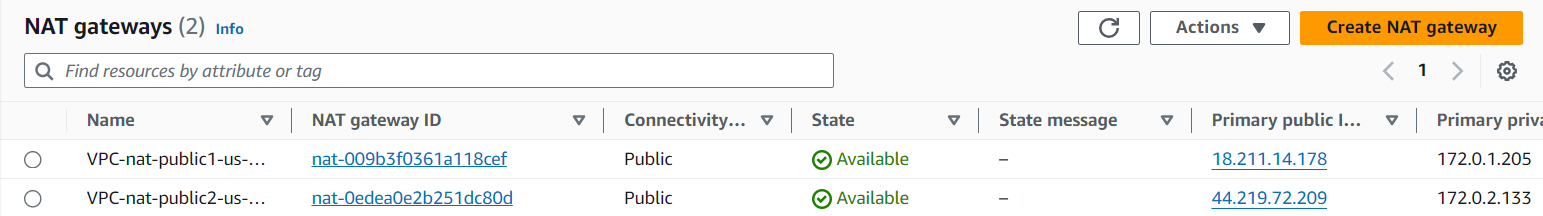




Navigate to **Internet Gateway** and confirm that Internet Gateway is attached to the VPC we just created.

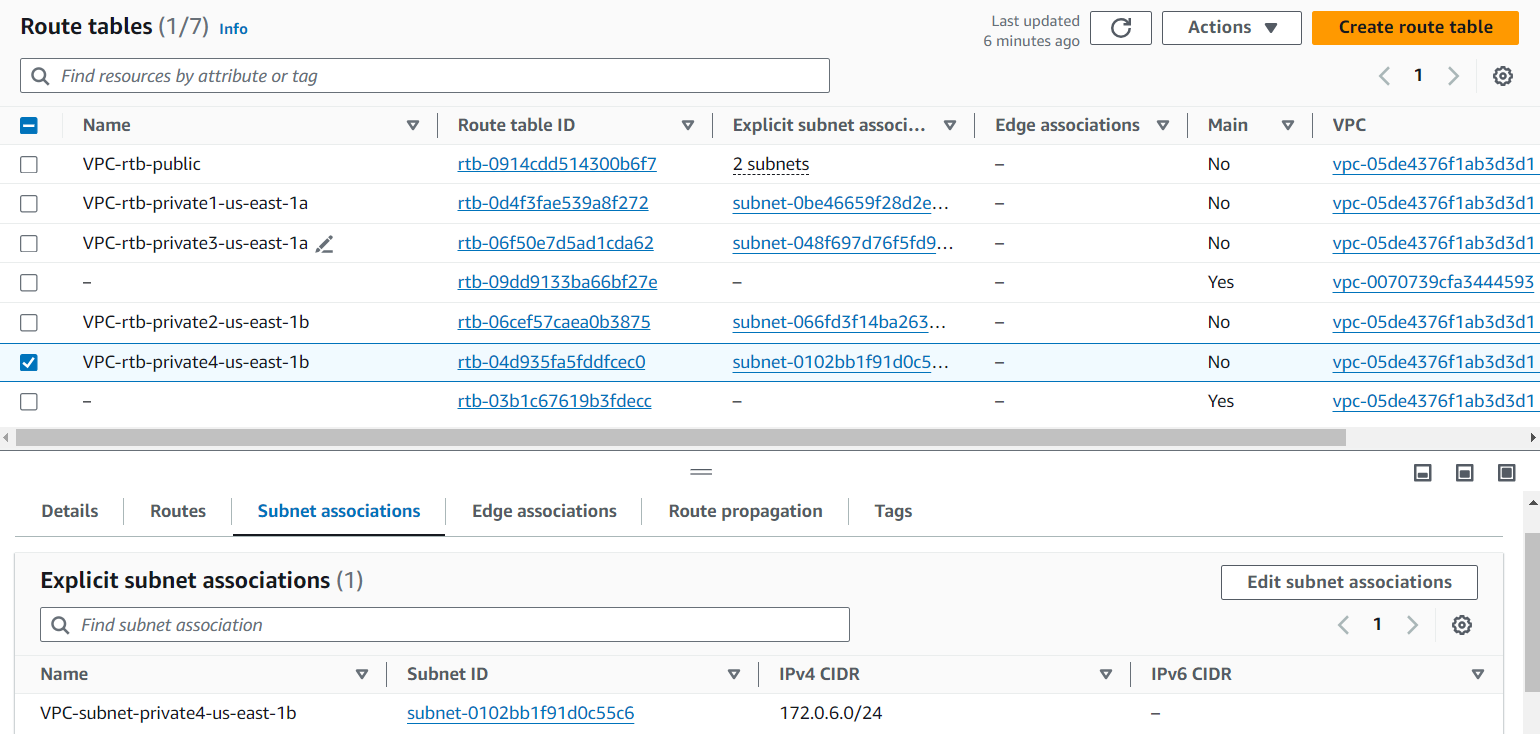


Navigate to **NAT gateways** and confirm that two NAT gateways are created.



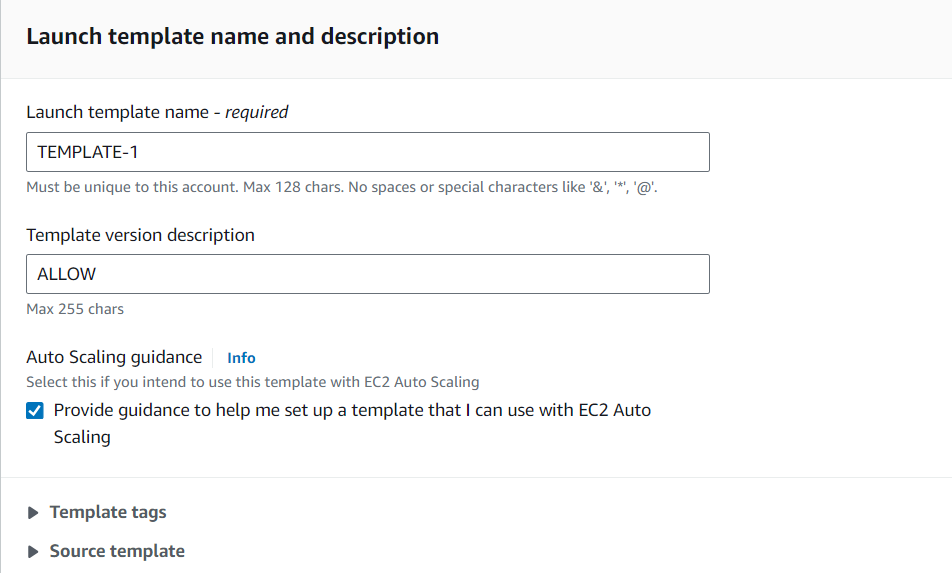
Navigate to public **Route tables** and confirm that both public subnets are explicitly associated. Confirmed that each private route table is explicitly associated with the private subnet.



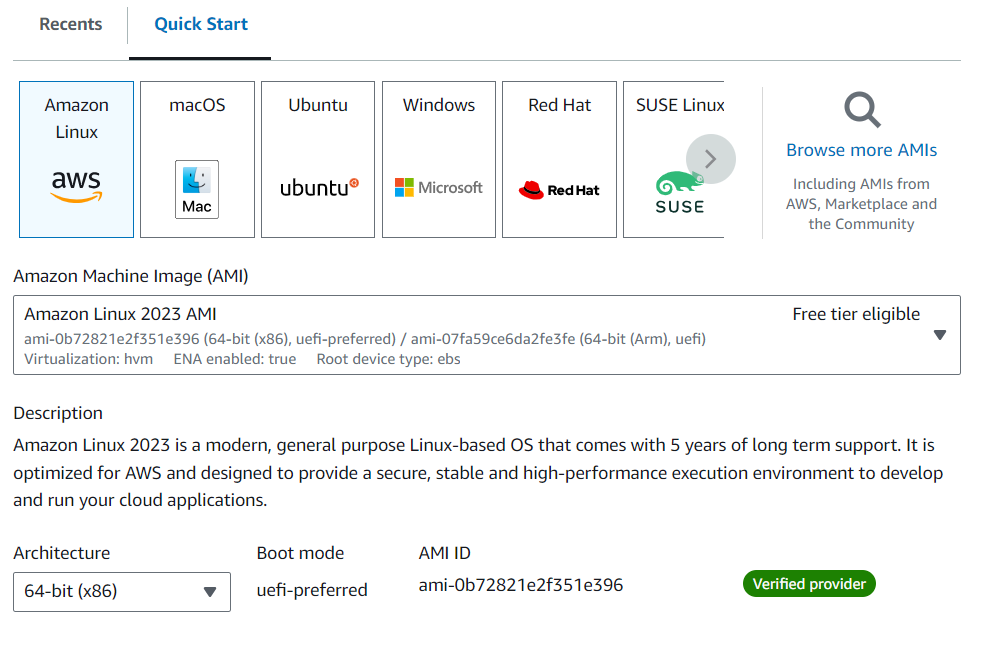


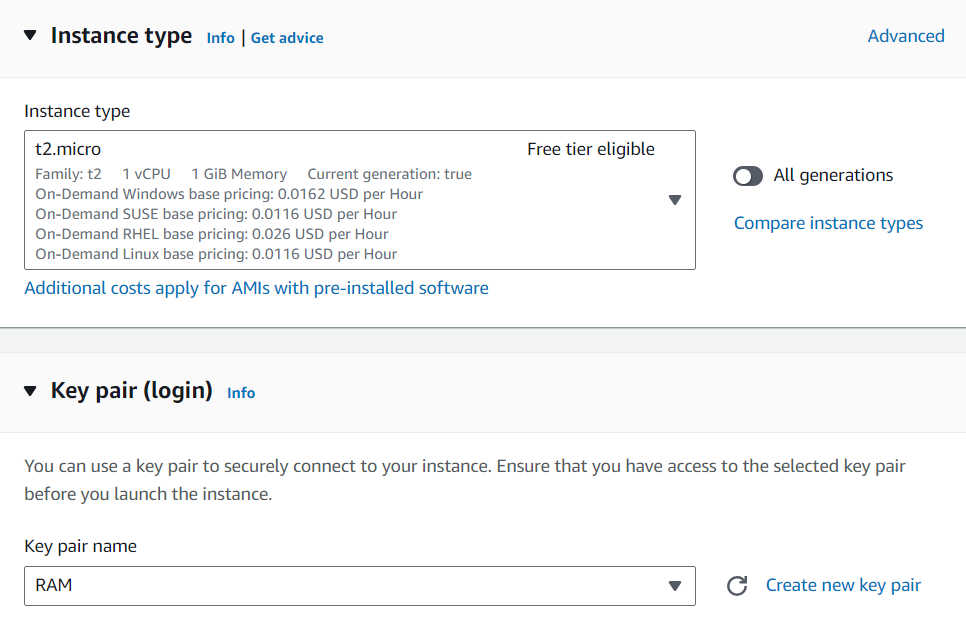
**STEP-1: CREATE WEB TIER**

Go to **EC2 > Launch Templates > Click Create launch template** button. Give the launch template name and description. Check the box for **Auto Scaling guidance**.

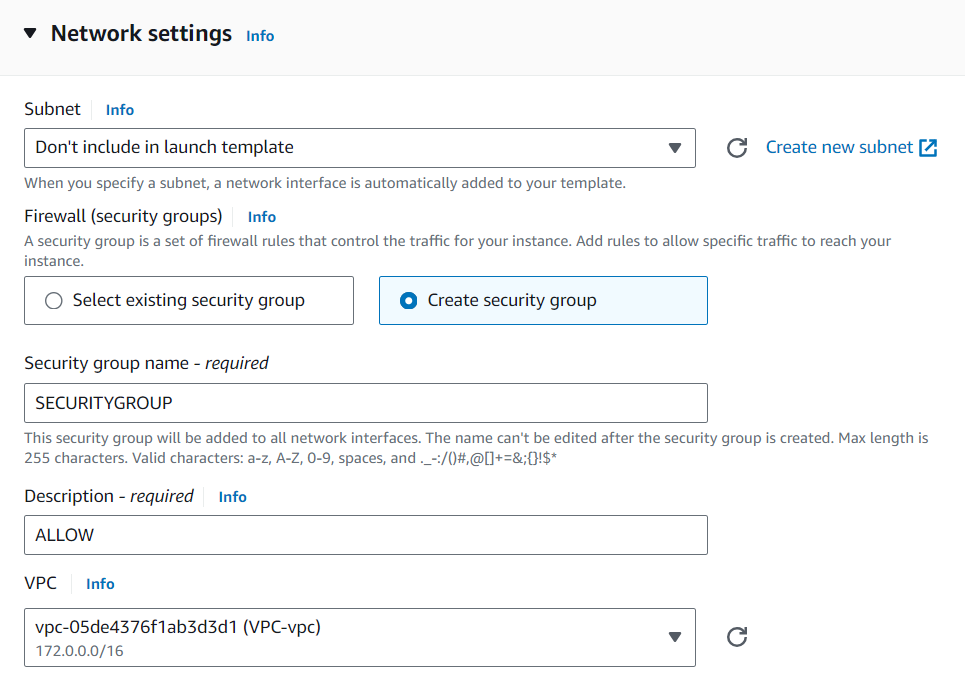


 Select your **AMI** and the **Instance type**. I selected Amazon Linux 2 AMI and t2.micro Instance type.





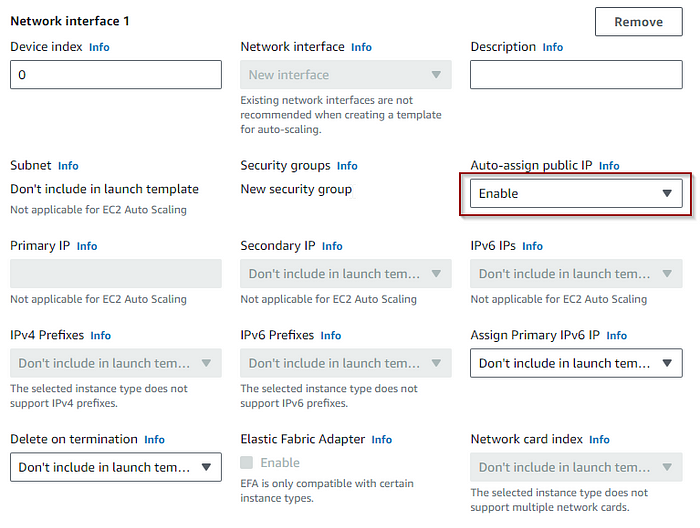
Select **Create security group** and give it a name and choose the VPC we created in Part 1.



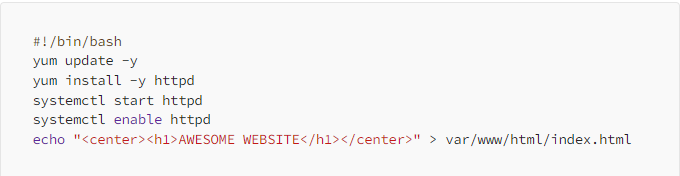
Click **Add Security Group Rules** button and add rules to allow ports for **ICMP, HTTP, and SSH** as shown below.



Under **Advanced network configuration**, enable **Auto-assign public IP**.



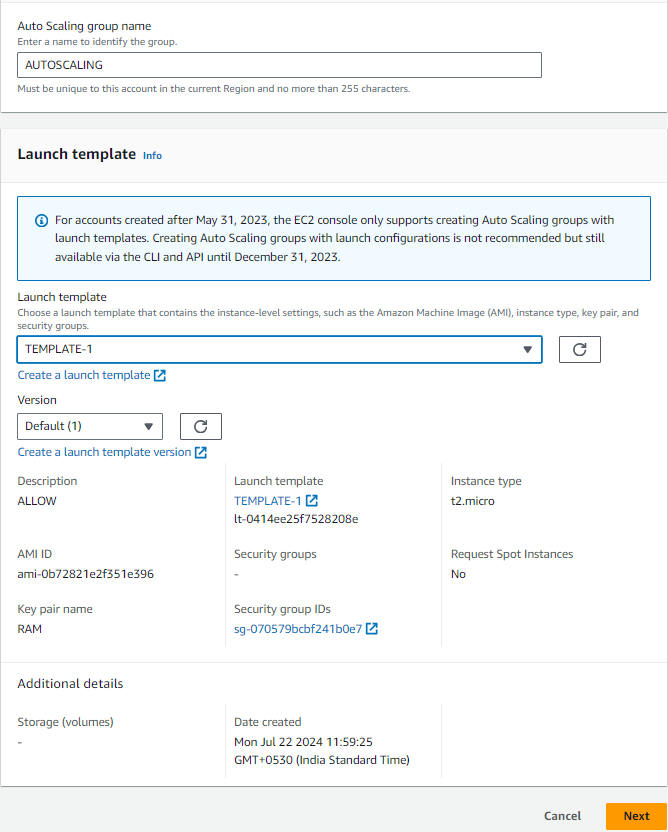
 Scroll to the bottom and add below script to the **User Data** field.

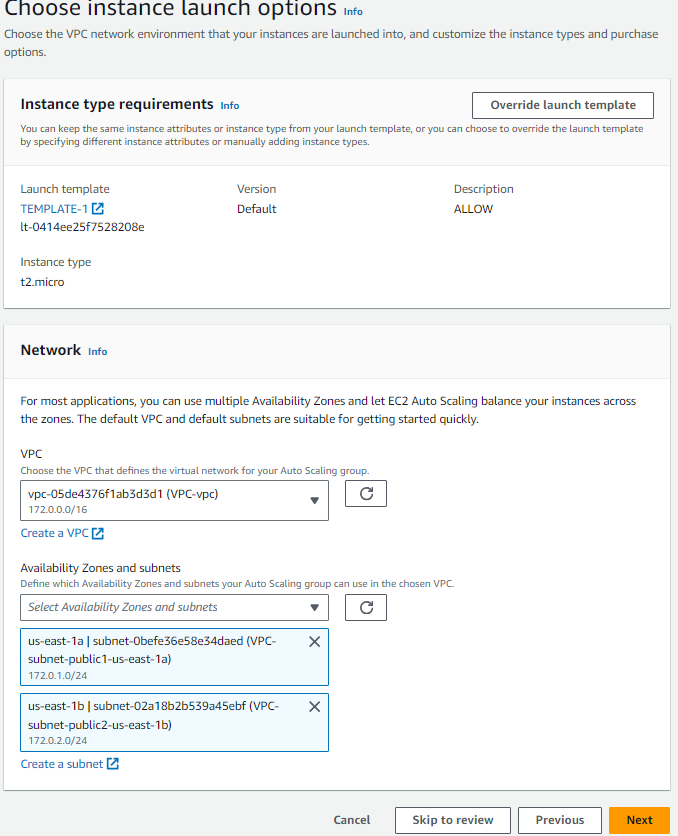


Click **Create launch template** button. Click **View Launch template** button.

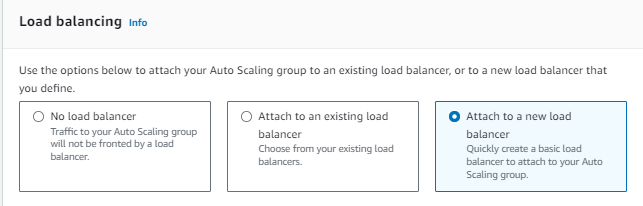


 Go to **EC2 >Auto Scaling Groups > Click Create Auto Scaling group** and give the **Auto Scaling group** name.

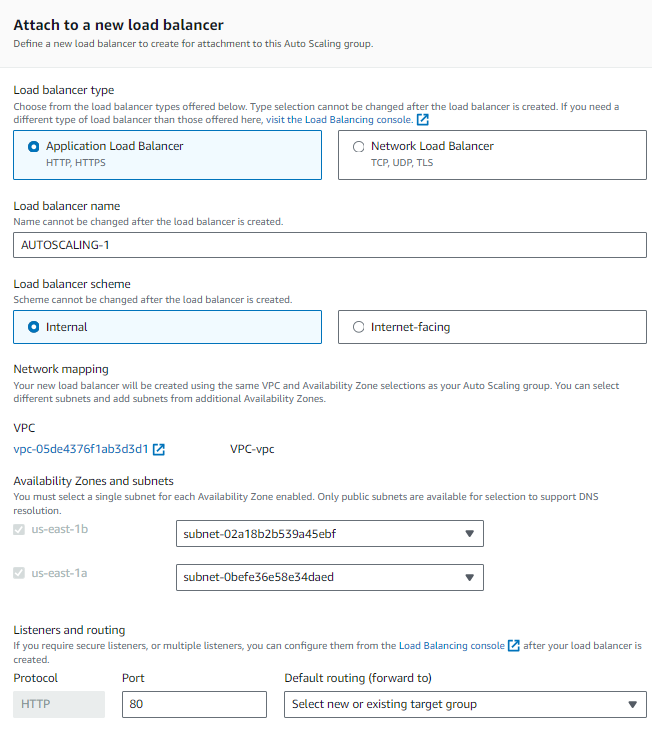




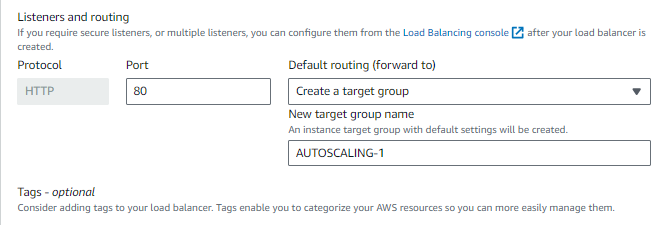
 Now it’s time to configure load balancer to distribute the traffic. Choose **Attach a new load balancer**.



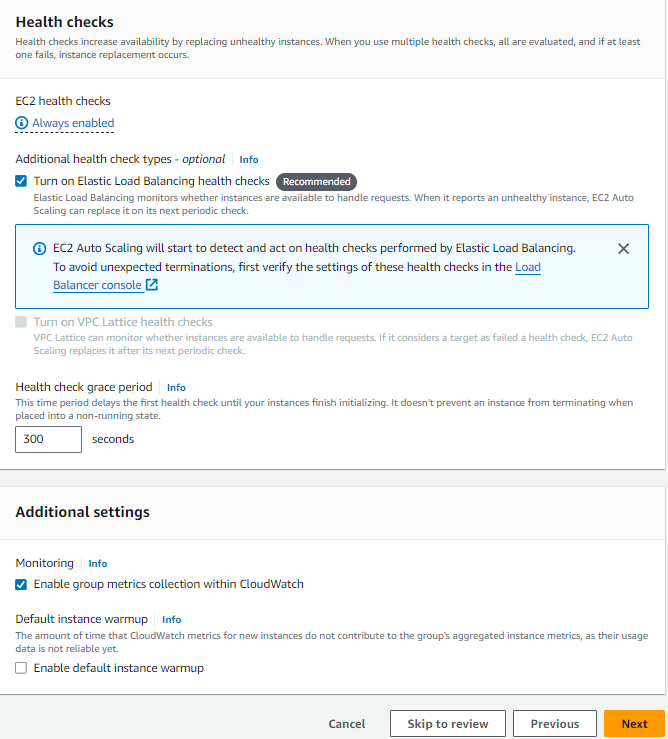
Choose **Application Load Balancer** type and make sure you select **Internet-facing Load balancer** scheme.



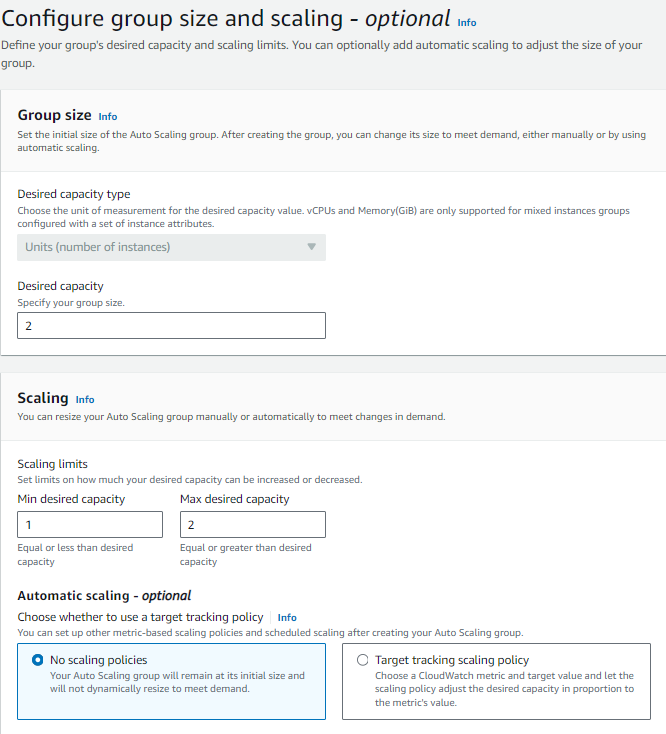
Select **Create a target group** for **Default routing**. A new target group with default settings will be created.



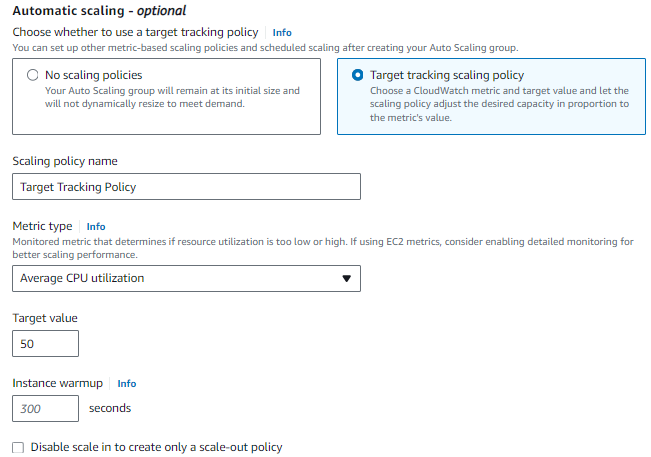
**Health check** is already enabled, and **Enable** **group metrics collections with CloudWatch**is already selected. Click **Next** button.

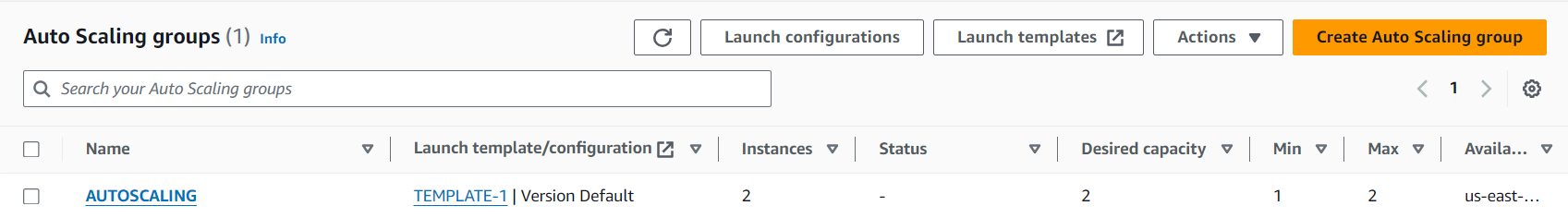


Specify the desired size of the Auto Scaling group.

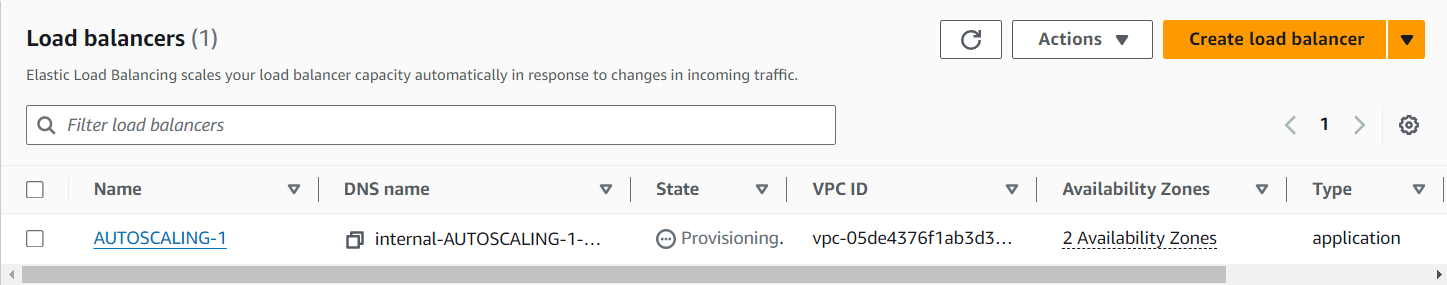


Choose **Target tracking scaling policy**. Click **Next** button. Click **Next** button again. On **Review** page, click **Create Auto Scaling group**.

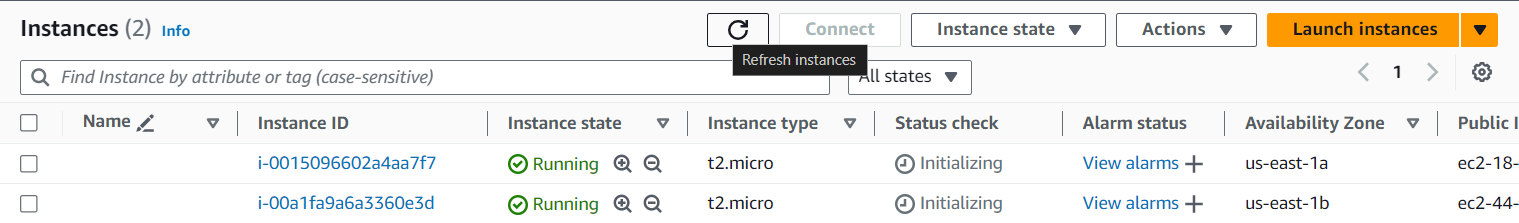




Go to **EC2 > Load Balancers** and confirm if it has been created successfully.

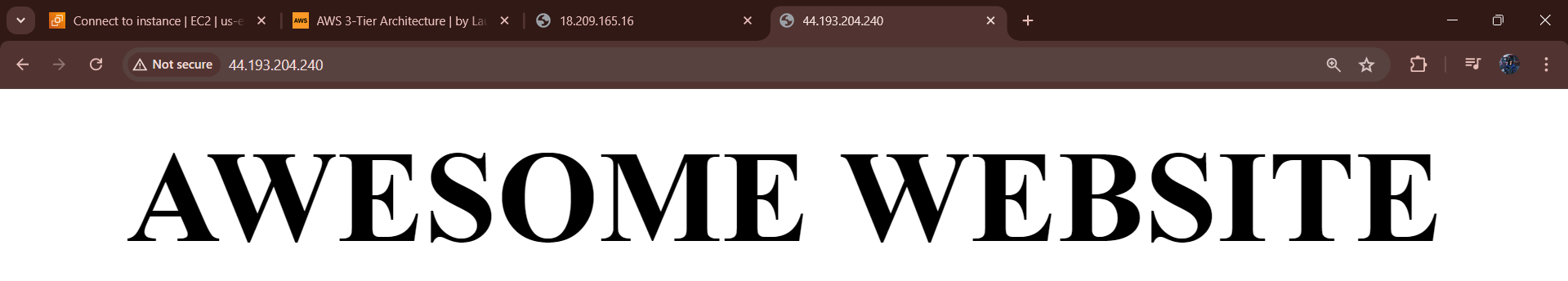


Go to **EC2** and confirm if two EC2 instances are running. Yes, two EC2 instances are initializing.



Go to a web Brower and enter the public IPv4 address. Yay! Success!

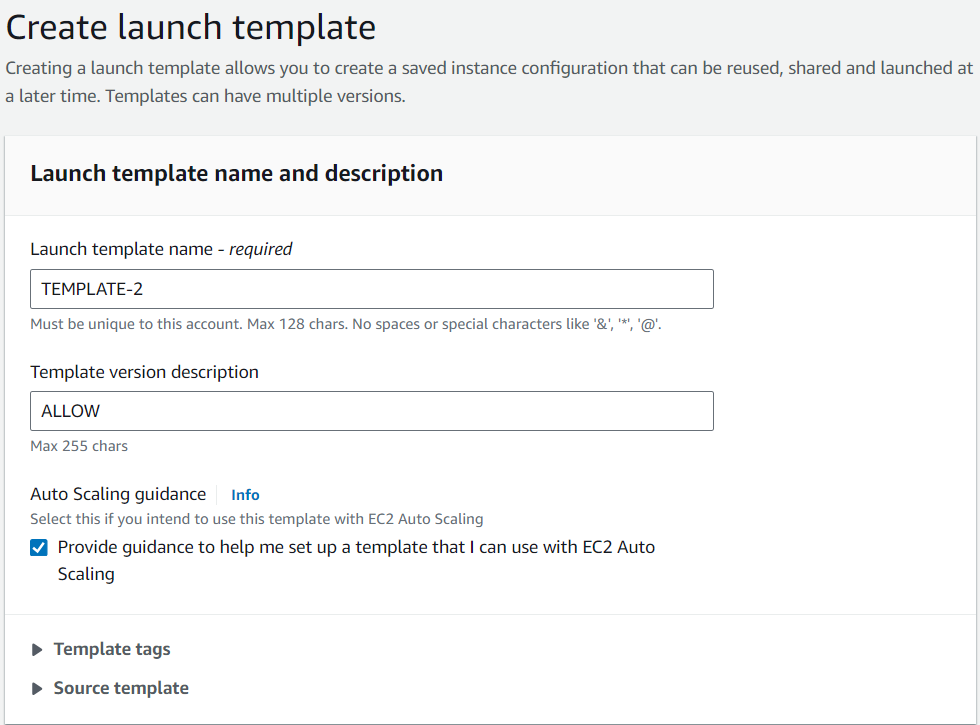
INSTANCE-1



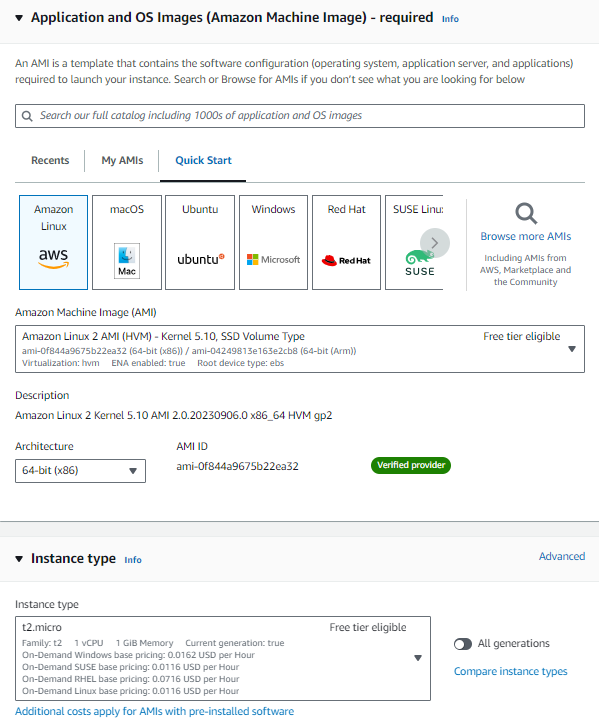
INSTANCE-2

**STEP-2: Create the App tier**

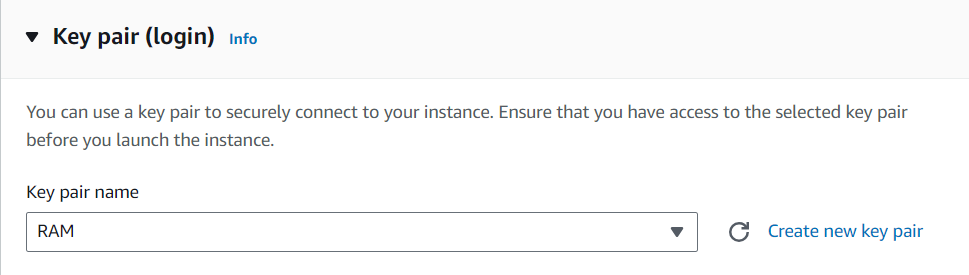
Go to **EC2 > Launch templates> Click Create launch template** button. Give the Launch template name and check for **Auto Scaling guidance**.



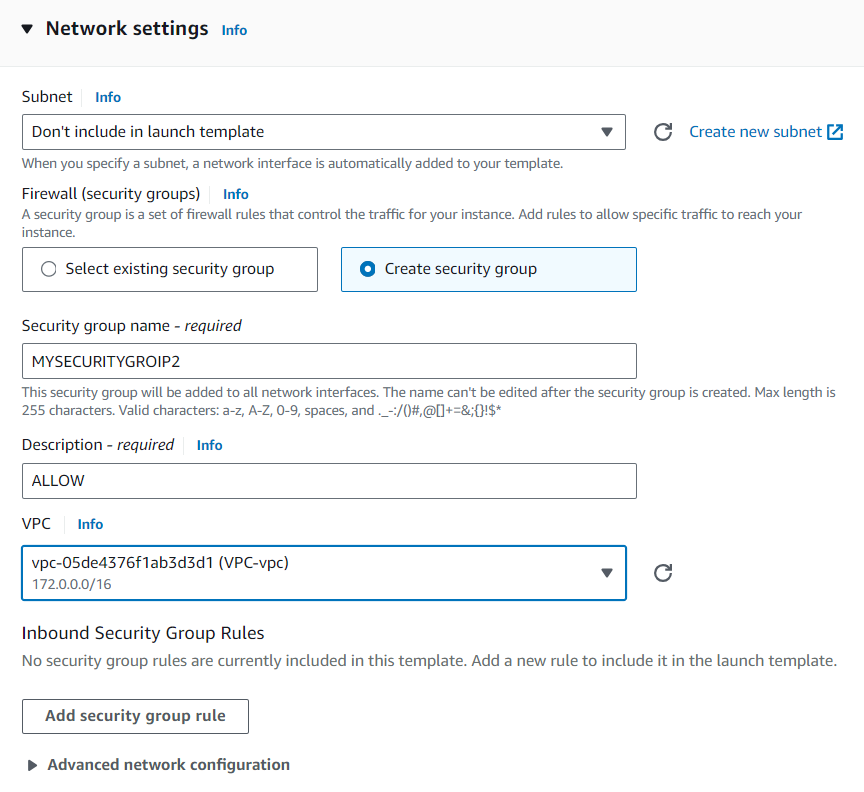
Select **Amazon Linux 2 AMI**and **t2.micro** Instance type.



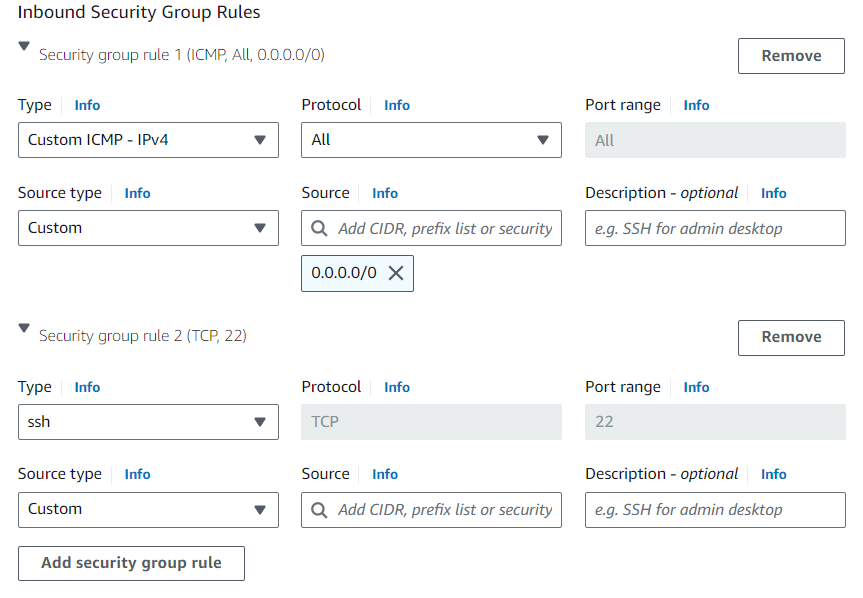
Select the K**ey pair** you already have.



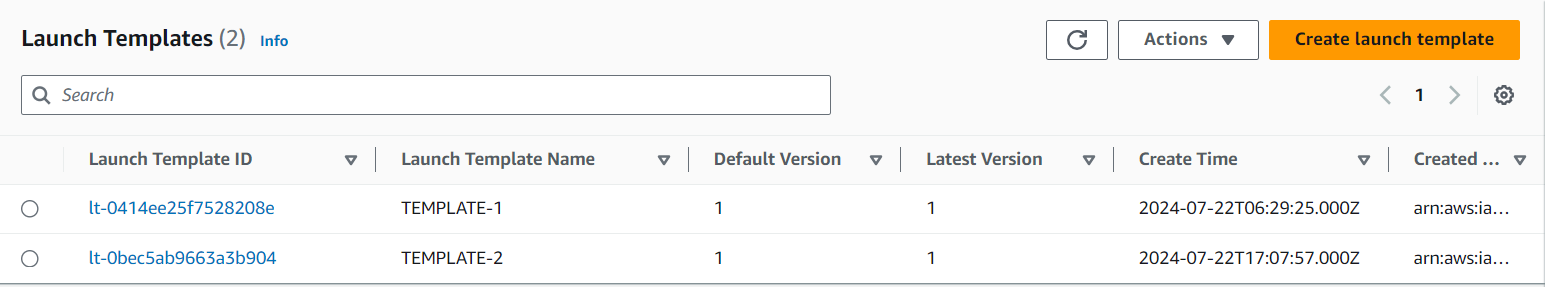
Under **Network settings**, select **Create security group** and give it a name and choose the VPC we created for tier 3 project.



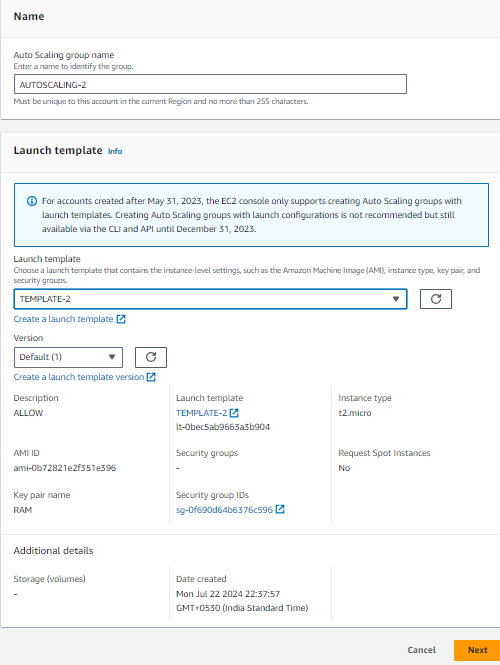
Click **Add security group rule** to allow ports for ICMP to ping the app EC2 via SSH. Choose Web tier security group as **Source** for ssh group rule.



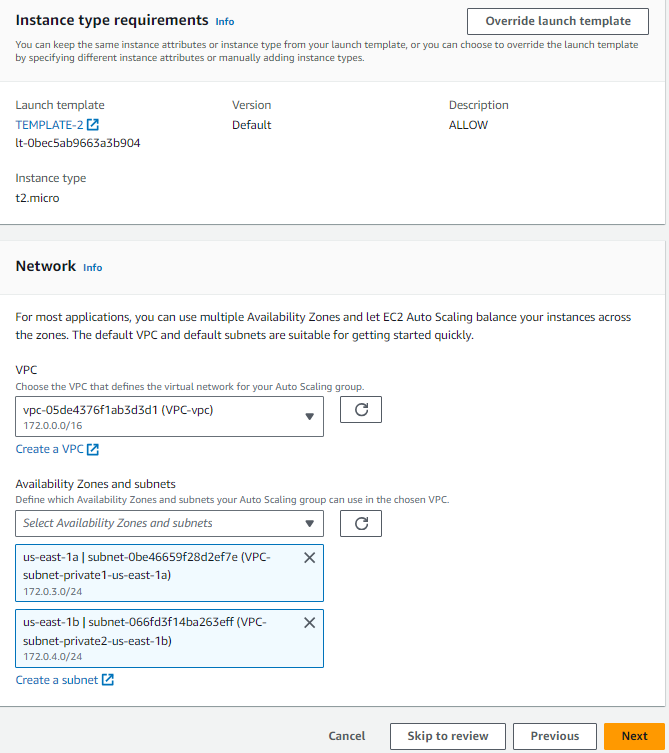
Click **Create launch template** button. And then, click **View Launch template**.



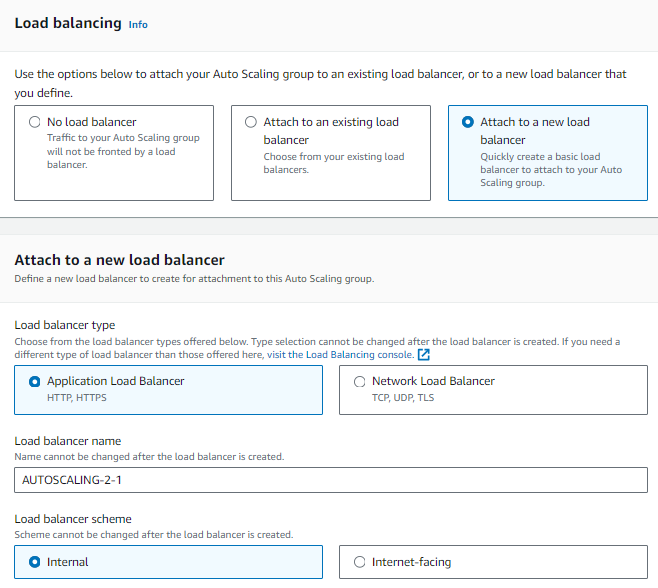
Go to **EC2 > Auto Scaling groups > Click Create Auto Scaling group** button. Give it a name and select the launch template for app tier and click Next button.



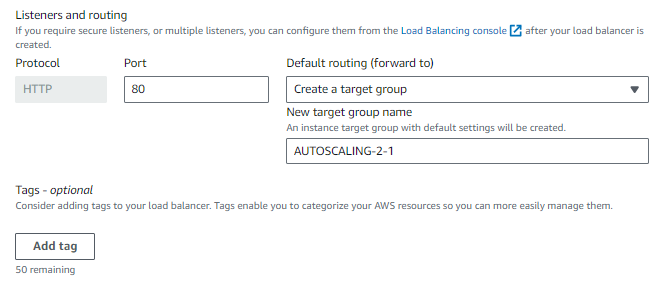
SelectVPC we created and select private subnets created for app tier. Click **Next** button.

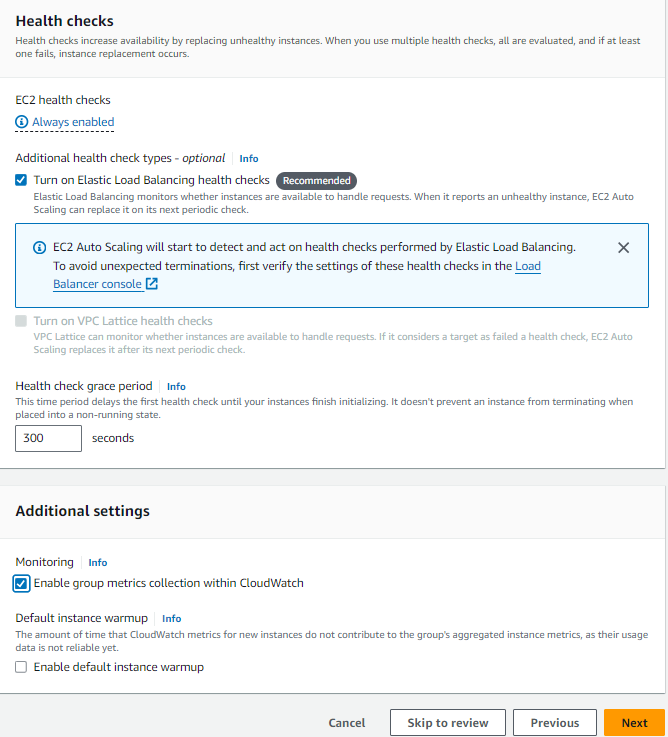


Choose **Attach to a new load balancer**. Then, select **Application Load Balancer**. For app tier, we need to select **Internal Load balancer** scheme.

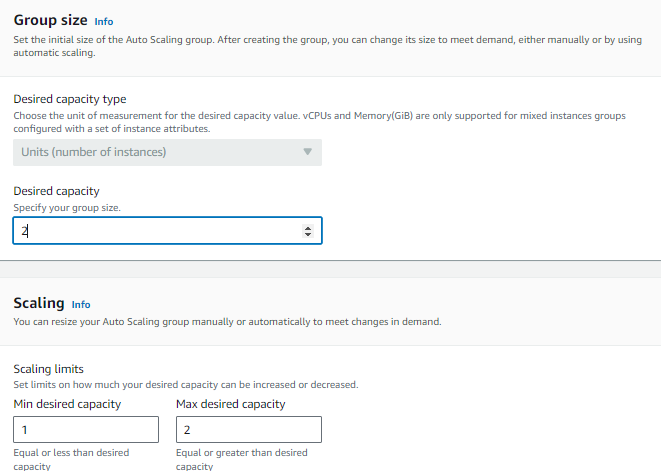


Select **Create a target group**. It will be automatically created for you. Keep the default settings for **Health checks** and click **Next** button.

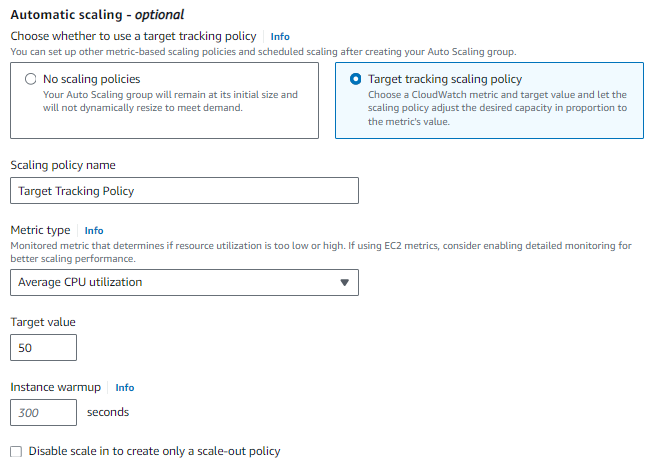




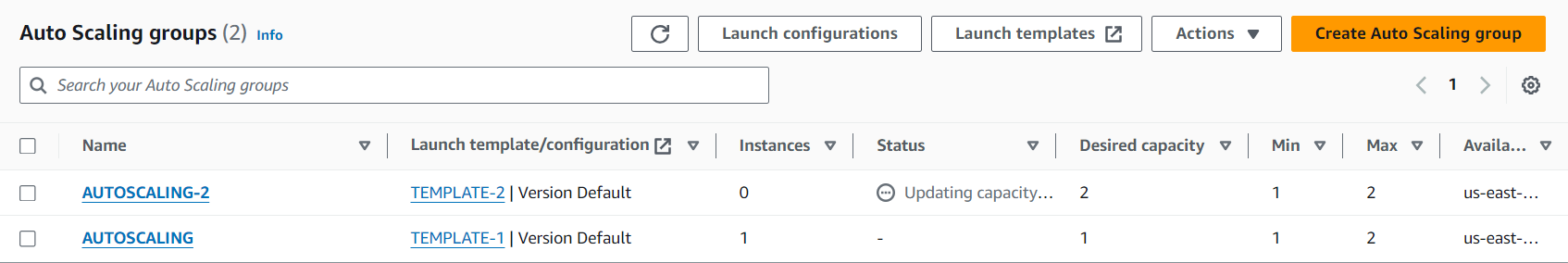
Enter your desired capacity.

****

Enter your **Scaling policies**. Then, click **Next** button.

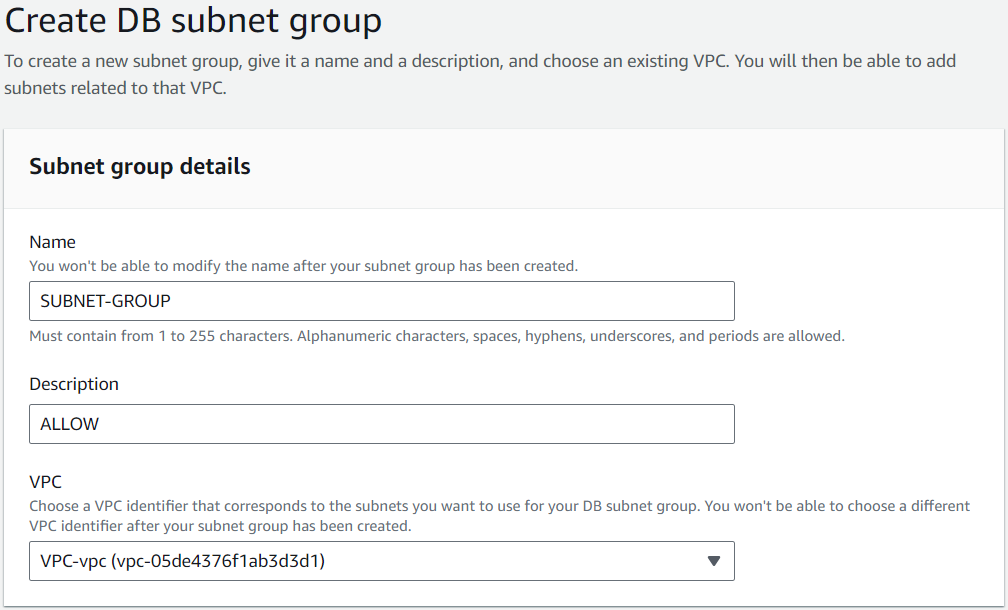


Review and then click **Create Auto Scaling group** button.

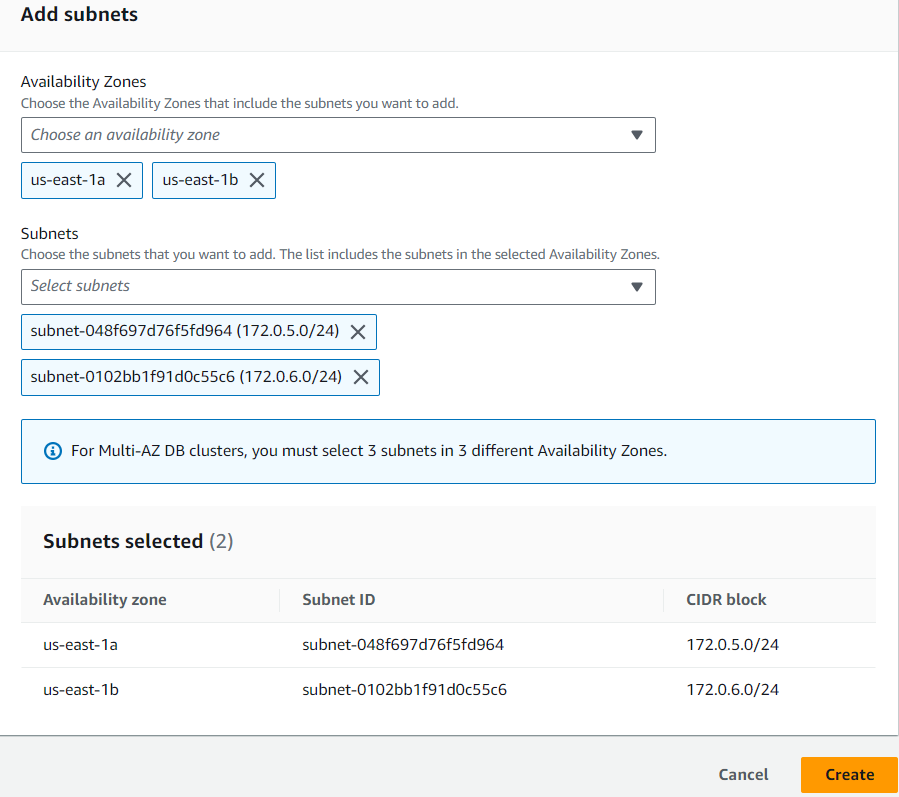


**STEP-3: Create the Database Tier**

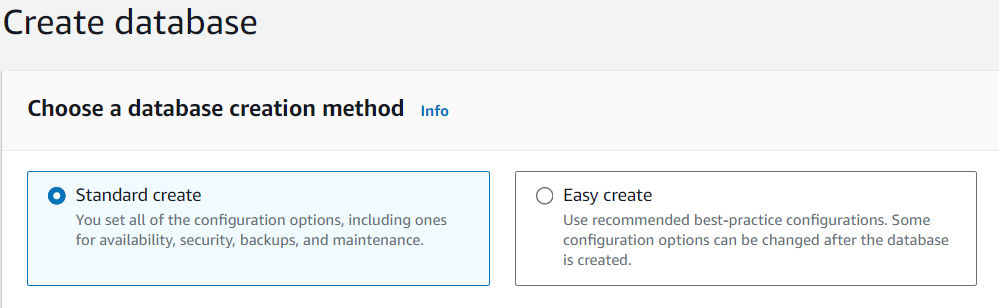
Navigate to **Amazon RDS> Subnet groups > Click Create DB subnet group** button. Give a name and choose the VPC we created.



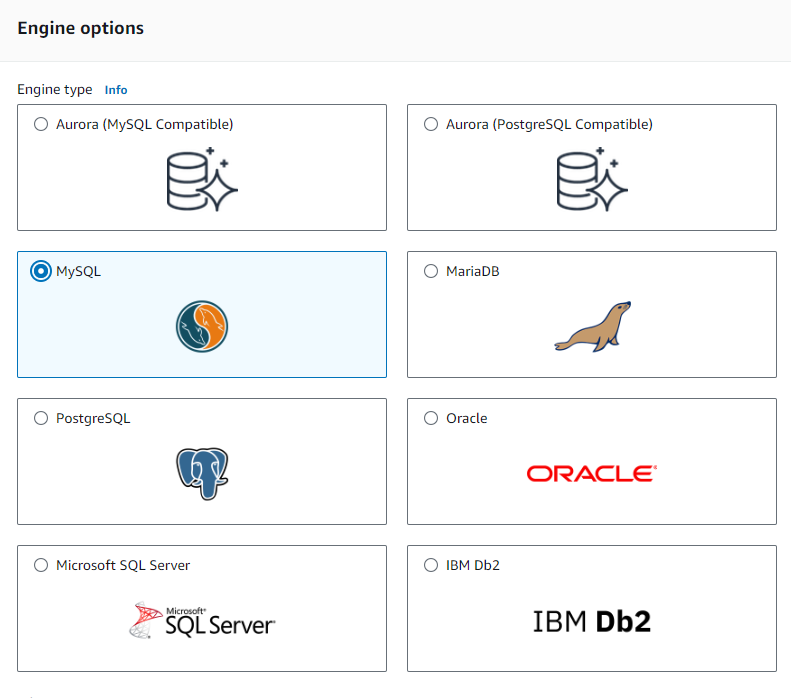
Add the availability zones we used for Web Tier and App Tier and select the two private subnets created for Database Tier. Click **Create** button.



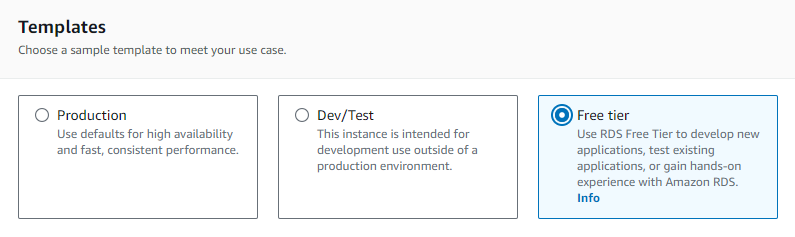
Navigate to **RDS > Click Create database** button. Choose **Standard create** method.



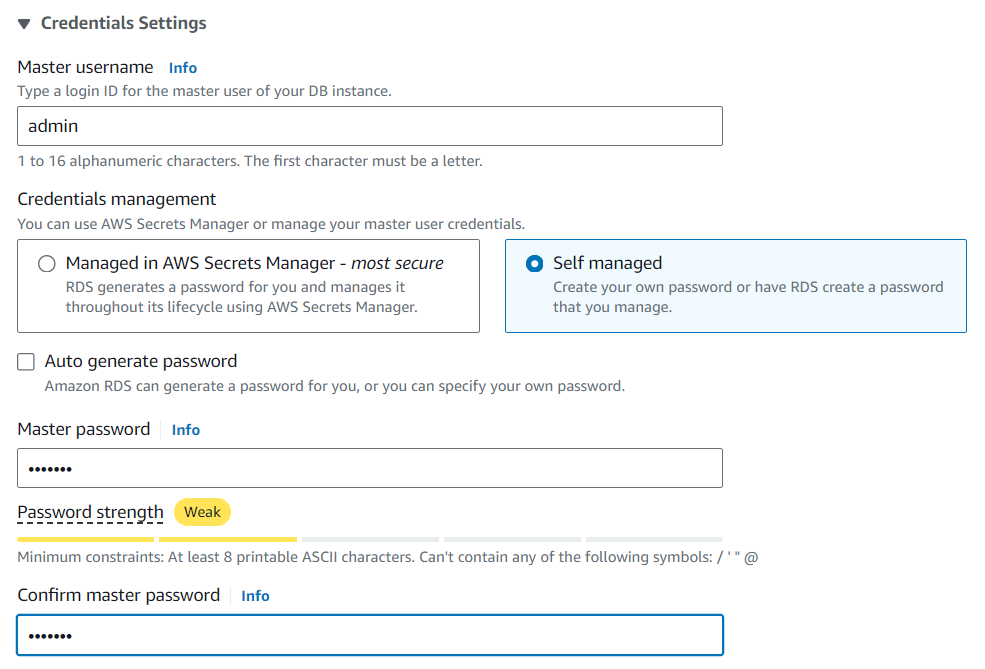
Select **MySQL** Engine type.



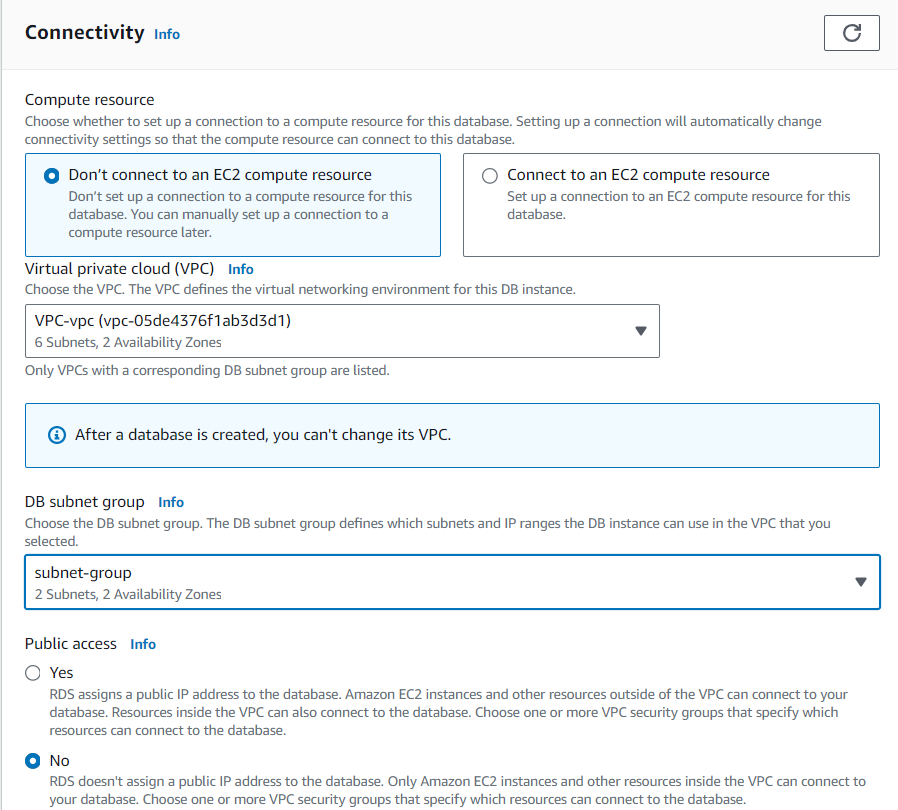
Choose **Free tier** Template.



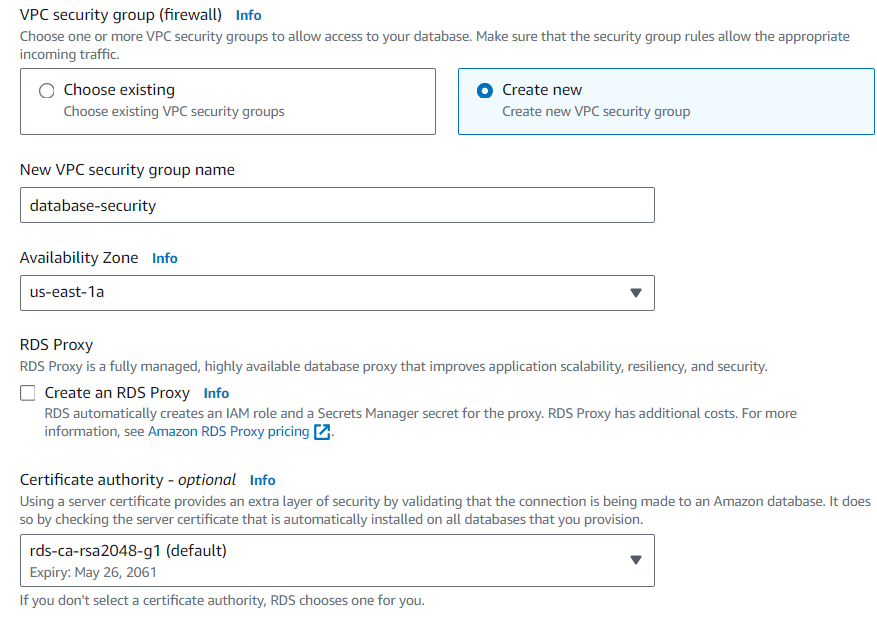
Give your DB instance name and set up the master credential.



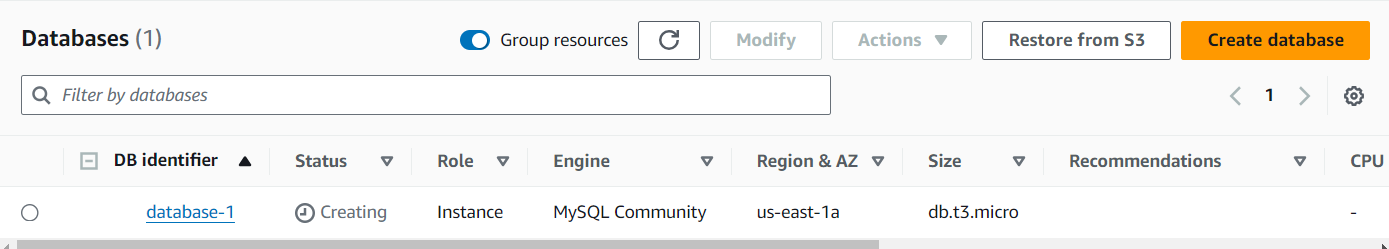
Keep the default Instance configuration and select **db.t2.micro** instance. Choose **Don’t connect to an EC2 compute resource** and add VPC we created. Choose the **DB subnet group** we created. Select **No** for **Public access** so that only EC2 can connect to your RDS database.



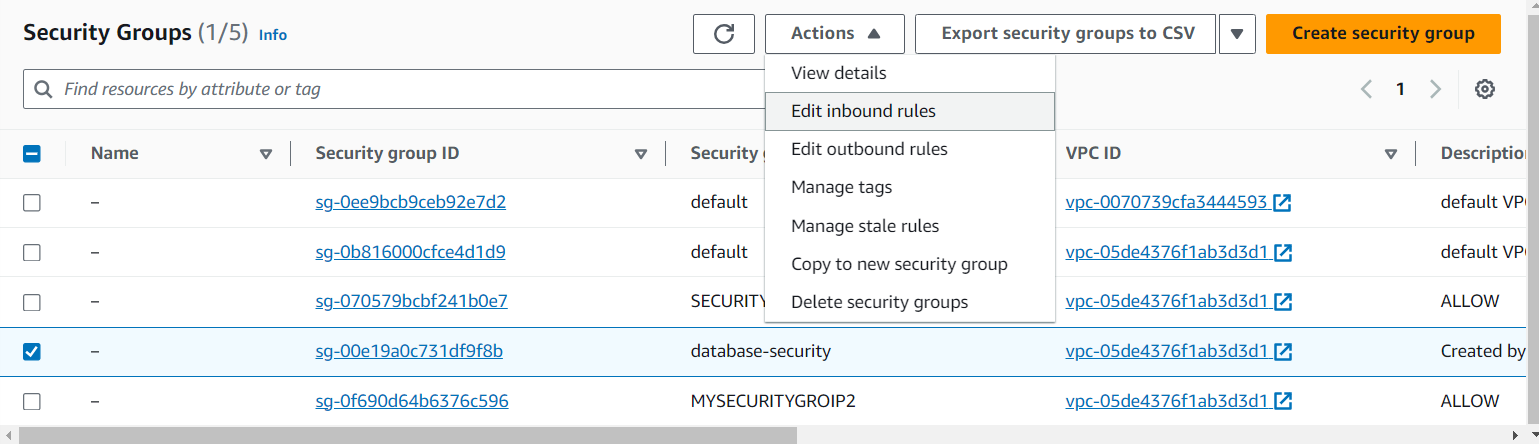
Create a new VPC security group. Give it a name and select us-east-1a for Availability Zone.



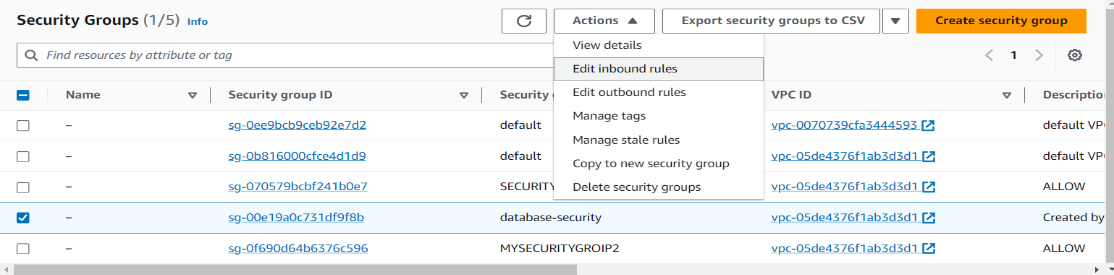
Keep the default setting for the rest of sections and then click **Create database** button.



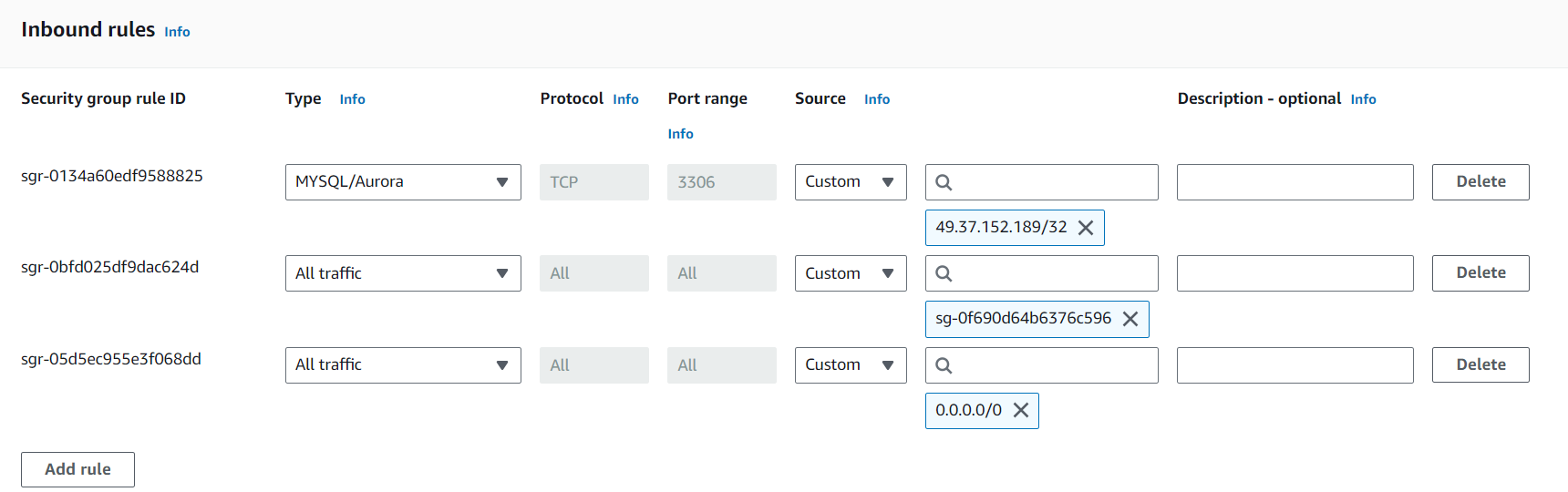
Go to **EC2>Security Groups** to verify the Inbound rules. Type key word data and the matched security group will show up. Select the security group.



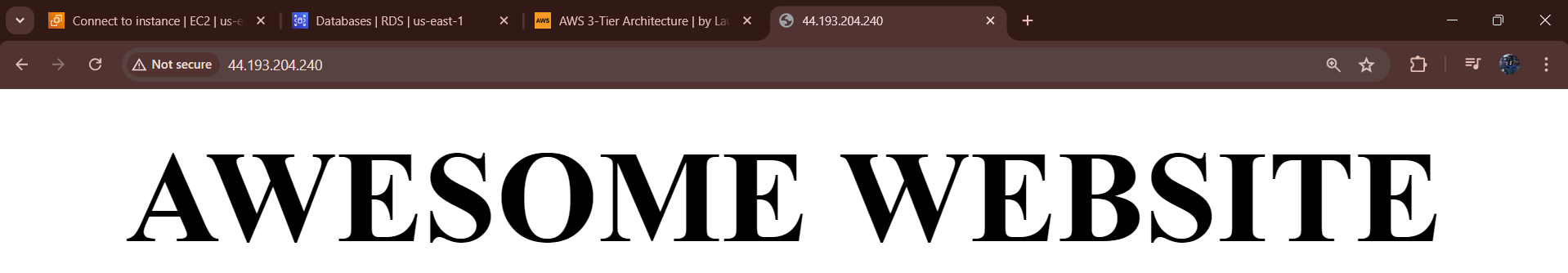
Go to **Inbound rules** tab. Click **Edit inbound rules**button.



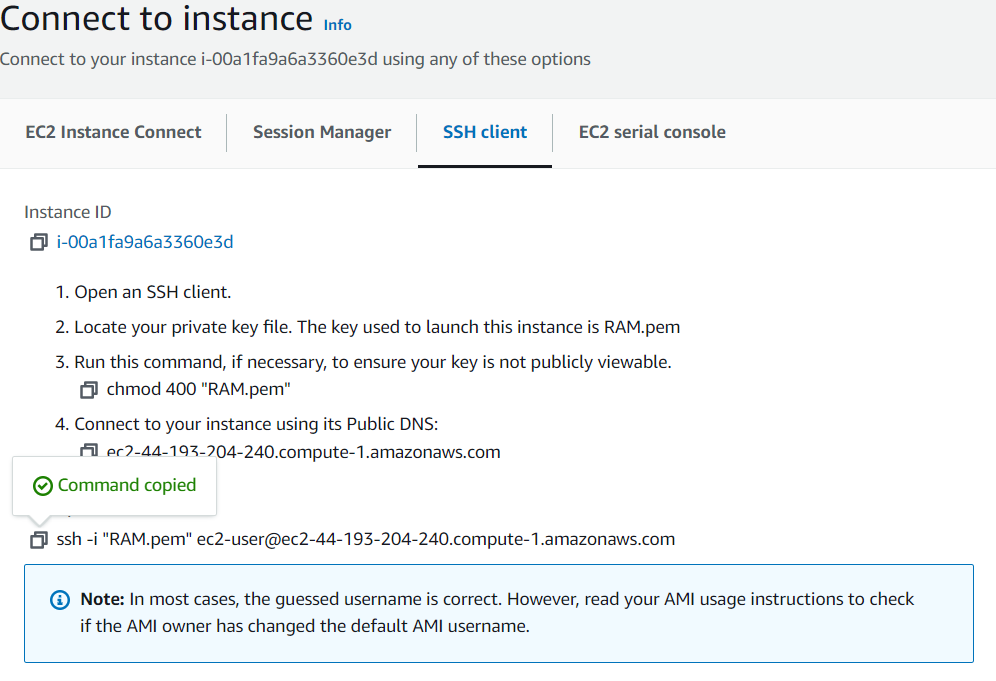
Change the Inbound **Source** to App Tier Security Group so that the App tier can communicate to Database Tier. Click **Save rules** button.



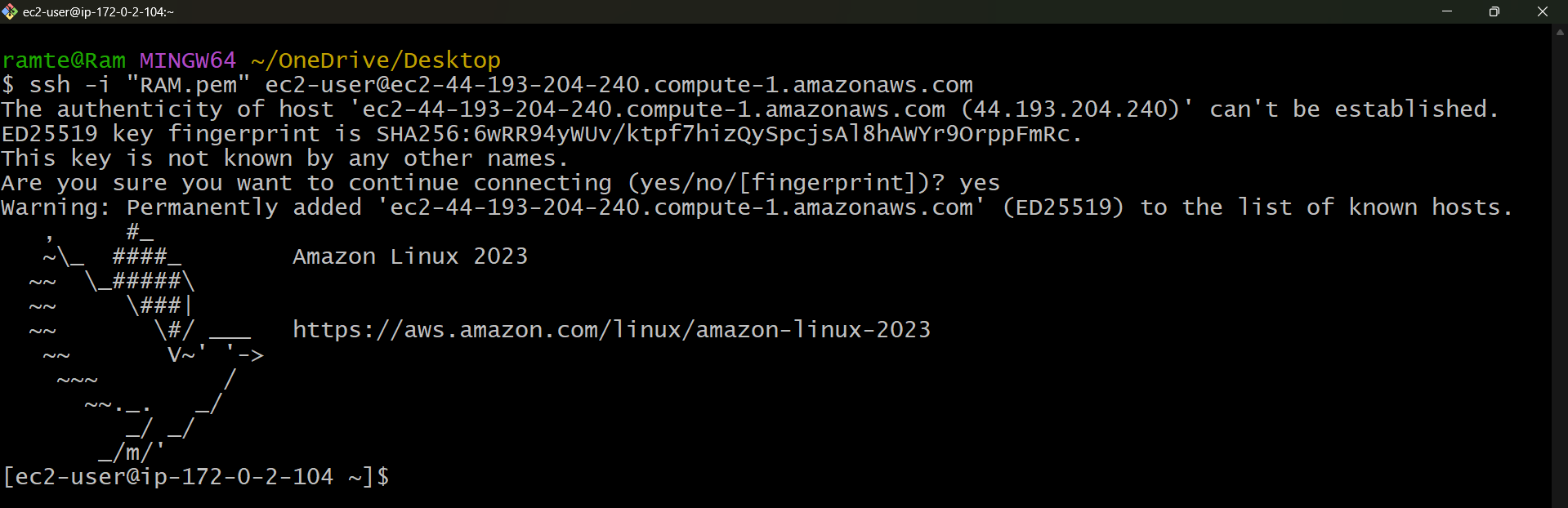
Navigate to a Web Tier EC2 Instances and grab the public IP address and enter it in the web address bar. Yes, success!



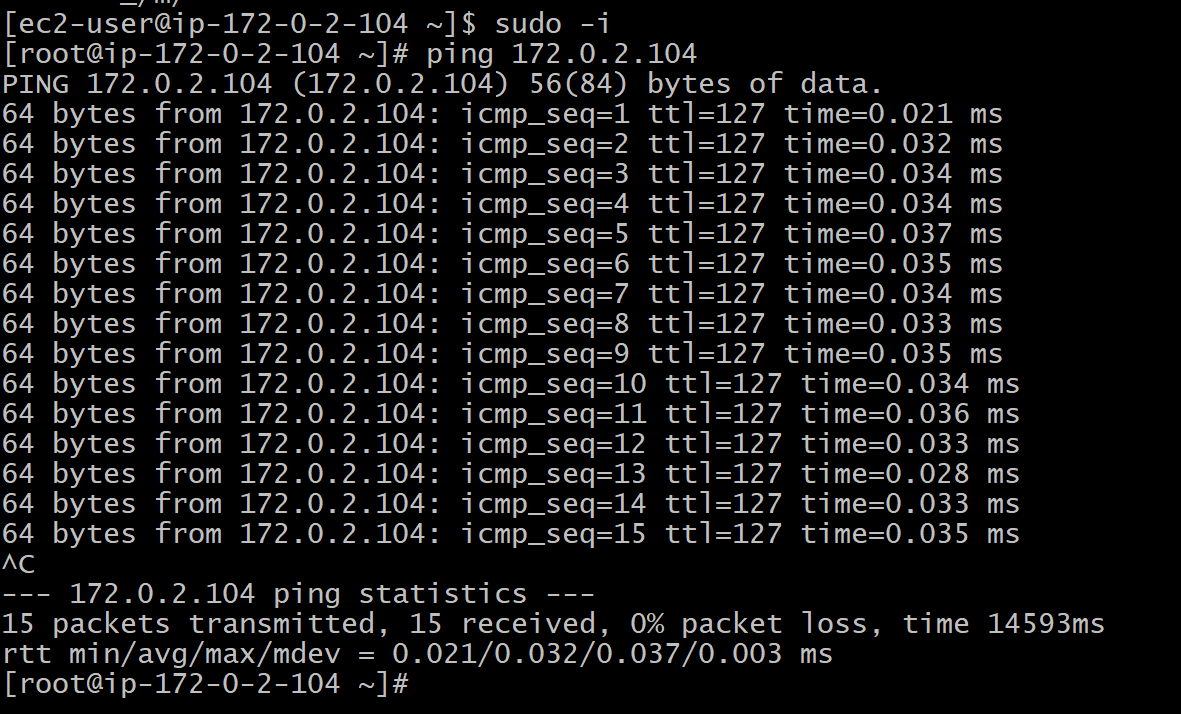
Now click **Connect** button to copy the ssh address.



SSH into the EC2 instance.



Ping one of the Private IPv4 from EC2 in the Private Subnet. Yes, I can ping the app EC2 instance private IP 10.0.3.180.



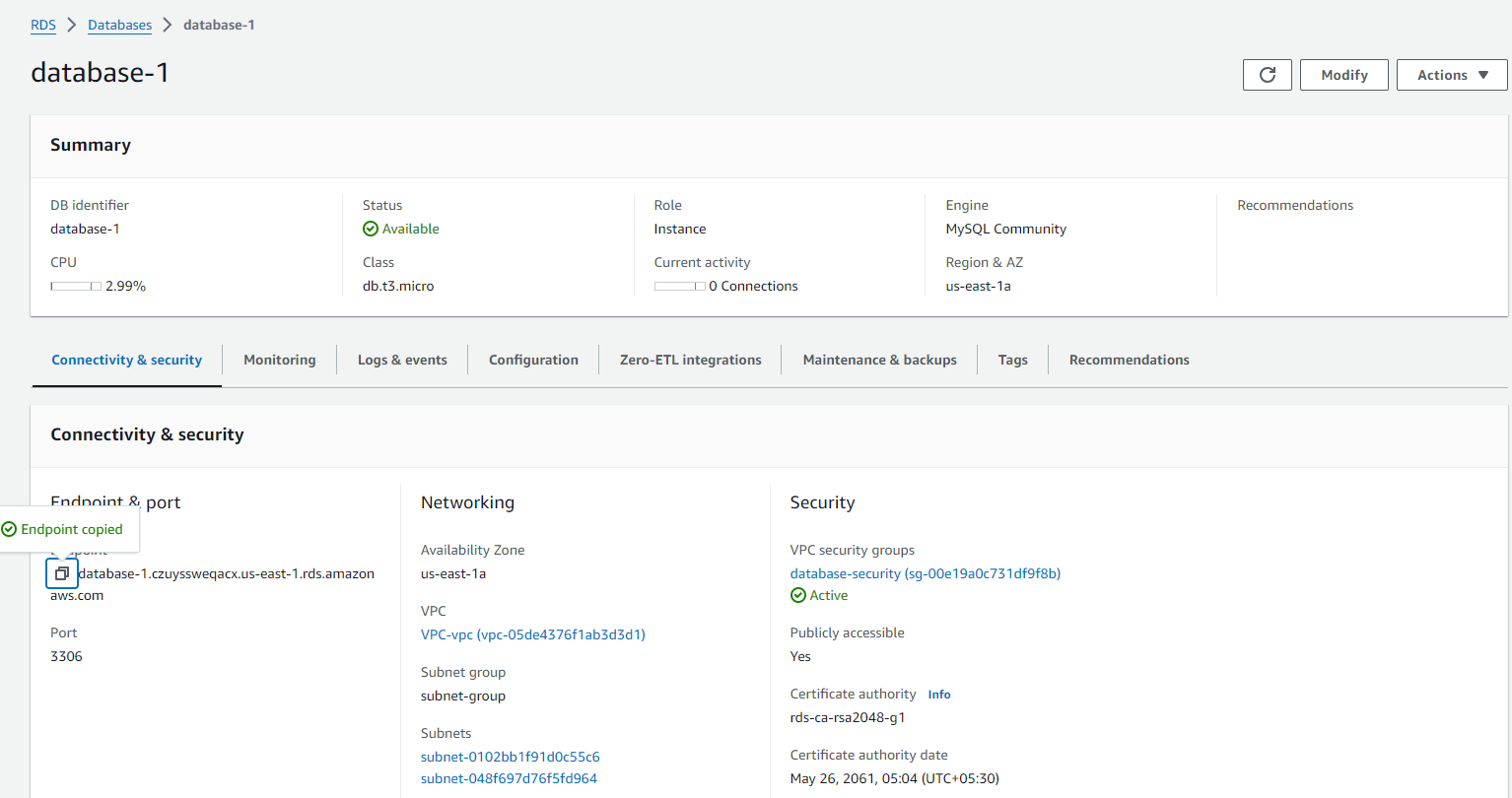
Now let’s test connectivity to the Database Tier. Run

#sudo apt update

#sudo apt install mysql-server

#sudo systemctl start mysql.service

Go to **RDS > Database > Tier 3 database > Click Connectivity & security tab** and copy the **Endpoint** address.



Run command below and then enter your admin password. Yay, success! Now, we are in the RDS database instance.

