## Name-:besta sarath kumar,

Mail:bskumar2927080@gmail.com

Batch-119,

7am batch,

## 

Three-tier architecture in AWS cloud refers to design pattern that divides an application into three logical physical computing layers. They are predominant software architecture for traditional client-server applications. Think of the application as having a three layered superhero team for your applications.

1. **Web Tier:** — the user interface and communication layer of the application where end users interact with the application. It collects information from the user interacting with your application. It’s like the stylish and interactive face of your creation! Whether it’s a snazzy web browser, sleek desktop app, or a fancy GUI, this tier has got it covered! HTML, CSS & JavaScript bring the magic to life here!
2. **Application Tier:** It is the heart of your app where all the behind the scenes action happens! In this tier, information collected in the presentation tier processed. Sometimes against other information in the data tier\_ using business logic a specific business rules. This tier processes the data collected from the Web Tier using its secret weapon, the “business logic” — a set of rules that makes things tick! It’s like the mastermind of the operation, orchestrating everything with programming languages like Python, Java, or even Ruby!
3. **Database Tier** — Stores and manages data application. Whether it’s a relational database like PostgreSQL or a cool NoSQL Database like Cassandra, this tier keeps all the valuable data safe and sound! All communication goes through the Application Tier. No direct chit chats between Web Tier and the Database Tier.

**Benefits of Three Tier Architecture**

The beauty lies in the ability to have separate development teams working on each tier simultaneously, skyrocketing your development speed and time to market! Embrace the latest and greatest languages and tools for each tier, and witness your app taking shape like never before! Plus, enjoy the freedom of scaling any tier independently as needed, ensuring improved scalability and enhanced reliability! And let’s not forget about security — with a well-designed application tier acting as a superhero firewall, malicious exploits like SQL injections don’t stand a chance!

**Picture this real world scenario**

Let’s say you’re building an e-commerce website! In this case, your website is a well oiled machine. It has three critical layers essential to its function. The first layer is the **“Presentation tier”**, it is what the customers see and interact with — it’s like the slick user interface that welcomes them to your online store!

The “**Application tier**” is the secret powerhouse behind the scenes! This layer processes all the orders, manages customer information, and handles all the business logic. It’s the brains of the operation, ensuring everything runs smoothly!

And lastly, we’ve got the “**Database tier**” — the vault where all the product details, customer data, and inventory are safely stored! Think of it as the backbone of your website, keeping everything organized and secure!

The beauty of the three-tier architecture comes into play here! Since each layer works independently, you can have different teams focusing on their specific areas. The frontend team can perfect the user interface, the backend team can tackle the complex business logic, and the database team can make sure all the data is managed efficiently!

Plus, as your e-commerce store grows, you can scale each layer separately to handle the increasing traffic and ensure a smooth shopping experience for all customers!

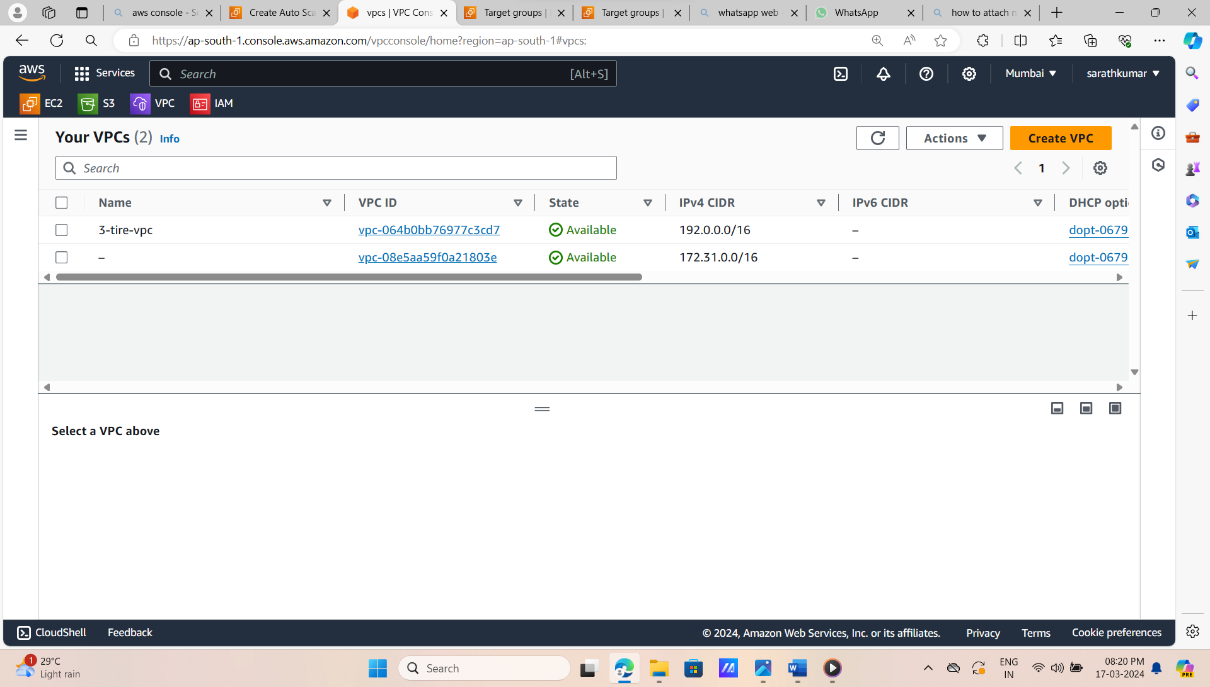
So, whether you’re starting a new online venture or revamping your existing store, a three-tier architecture is the way to go! It guarantees a reliable, secure, and high-performing website that keeps your customers coming back for more!

**Setting up the architecture**

**Step 1: Create a VPC with** **2 Public subnets and 4 private subnets across two availability zones.**

**Region-mumbai,**

**Vpc-192.0.0.0/16 ,**

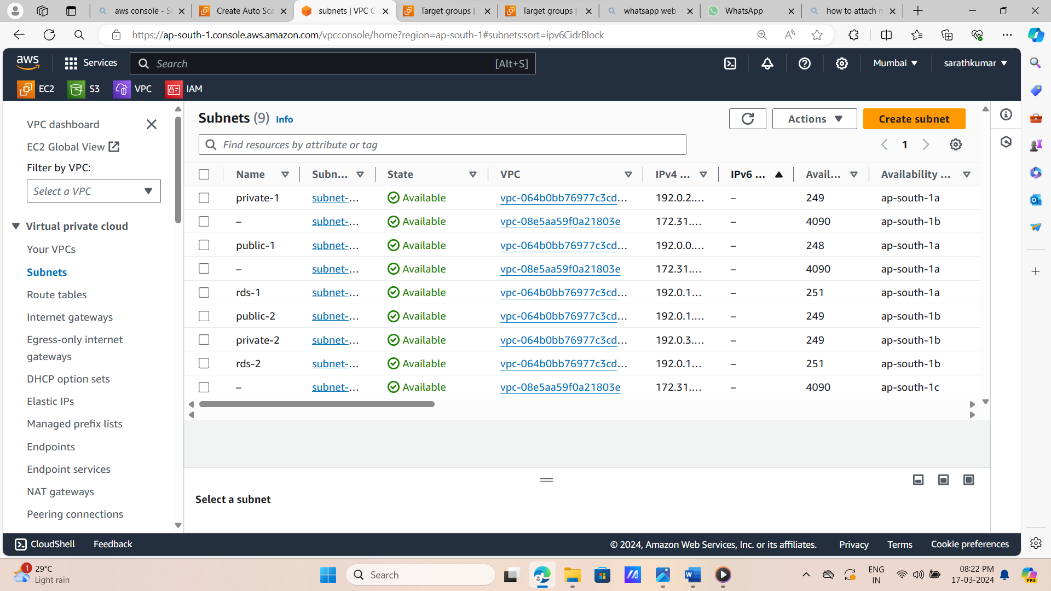
****

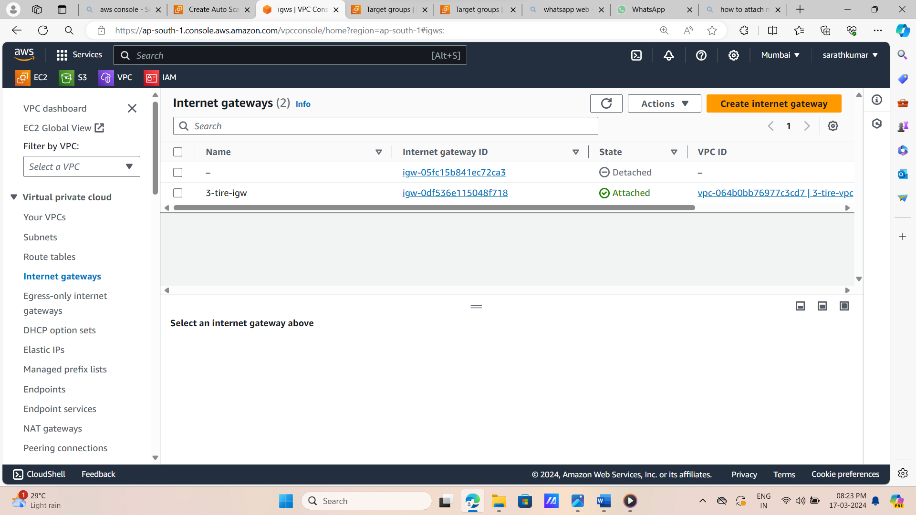
**Az----- Ap-south-1a ap-sout**

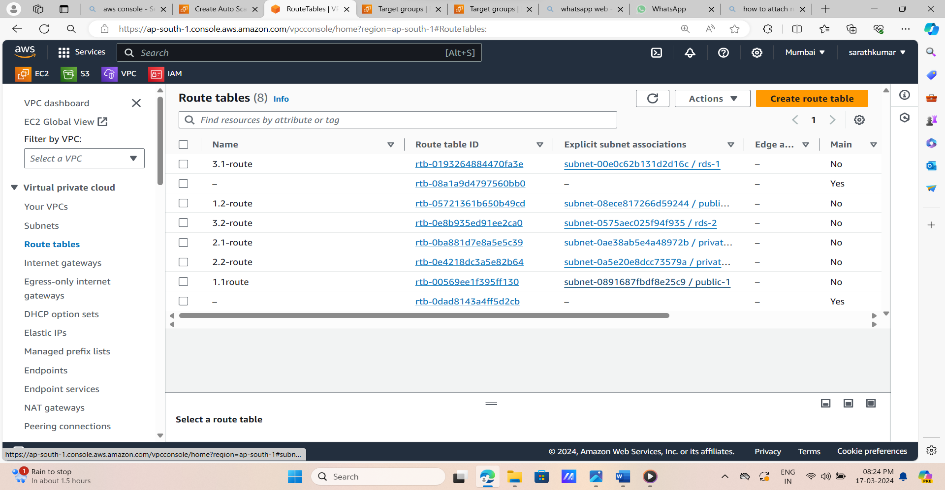
Public subnet ipv4---192.0.0.0/24 192.0.1.0/24

Private subnet ipv4—192.0.2.0/24 192.0.3.0/24

Rds-private sub ipv4-192.0.10.0/24 192.0.11.0/24



2.now create igw,and attach the vpc.

3.create route table each subnet. 

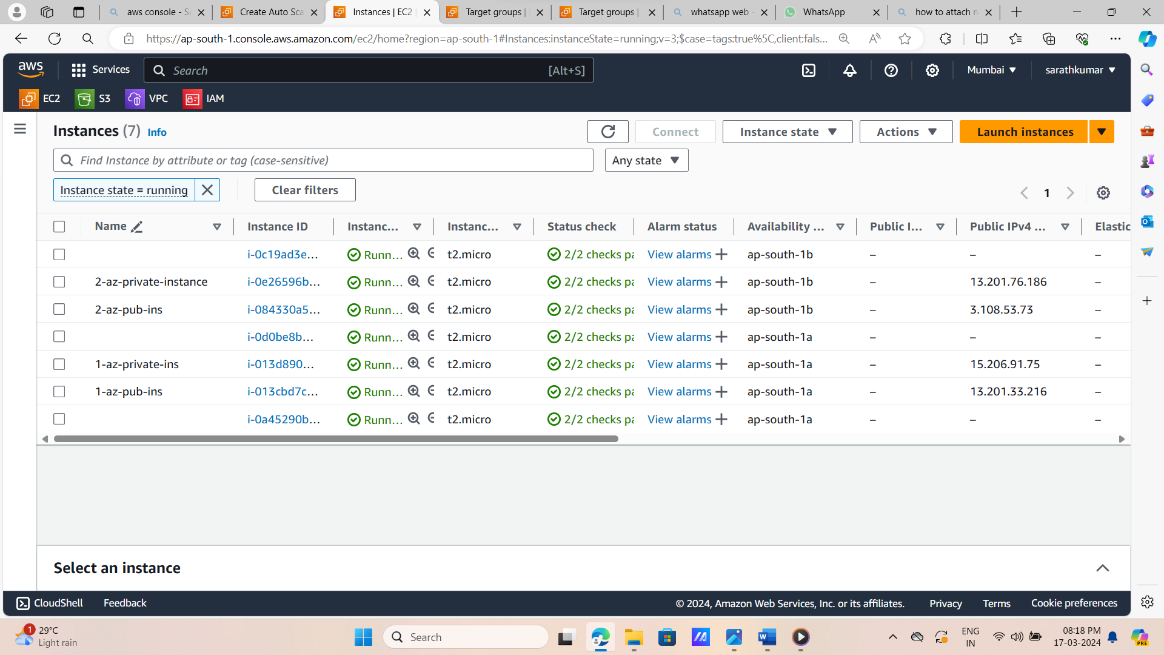
After create route table,association their subnets.

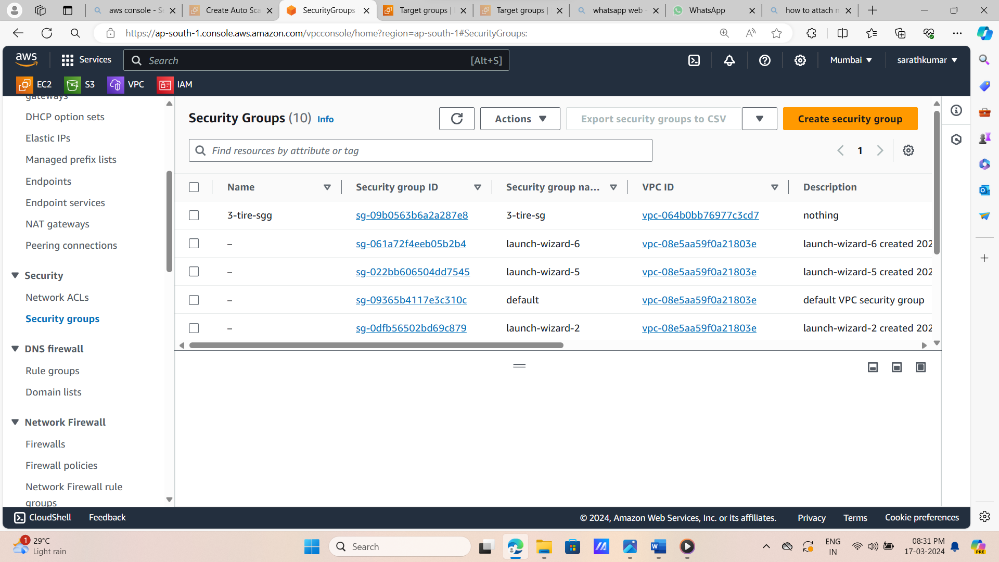
4.now create 4 instances with assign scecurity group,

2 public instances,and 2 private instances and there named assigned as

1-az-public instance, 1-az-private instance

2-az-public instance, 2-az-private instance

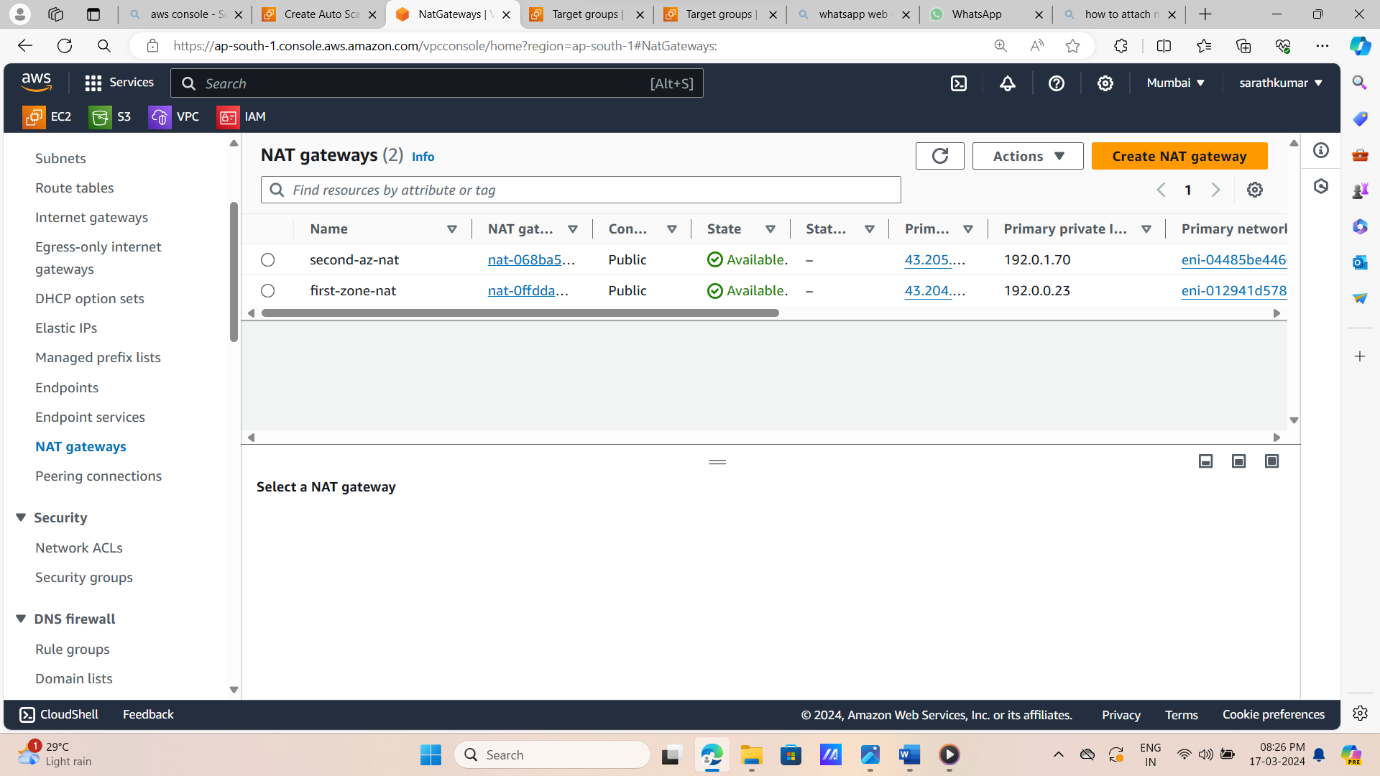
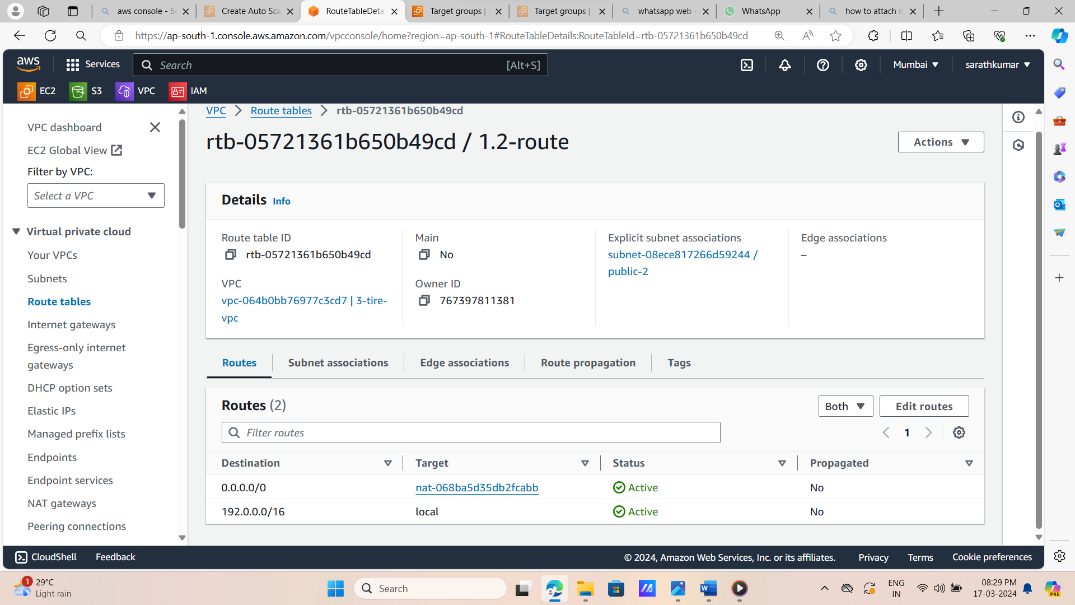




And launch the instances,and install nginx,and check the communication in private subnet.but there is no communication in public subnet to private subnet.atleast we want one way communucation.

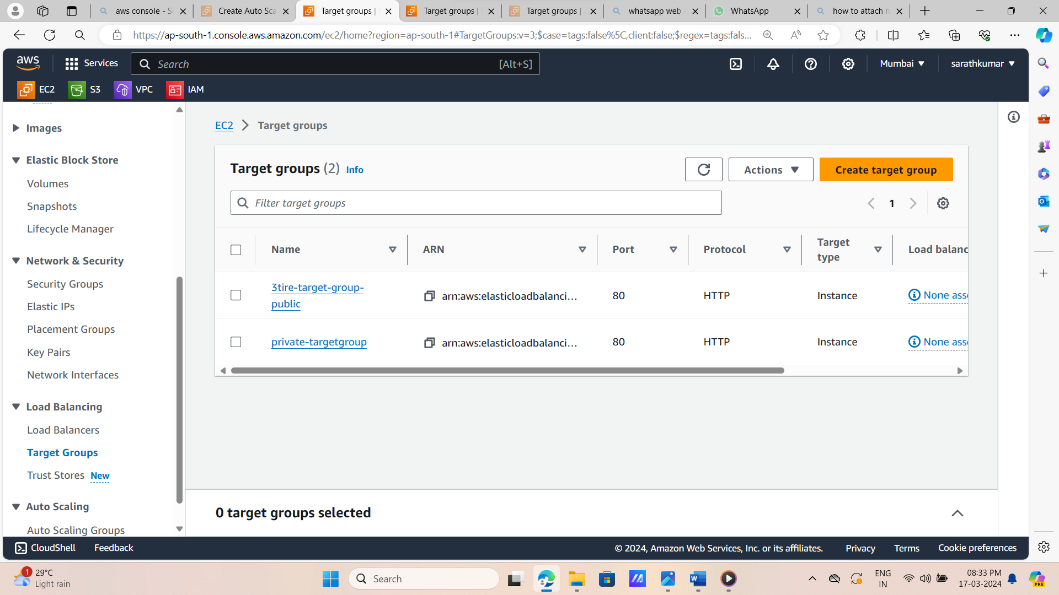
atleast one way communication purpas we create a nat gate way.in two different azs

5.create a nat gate way.

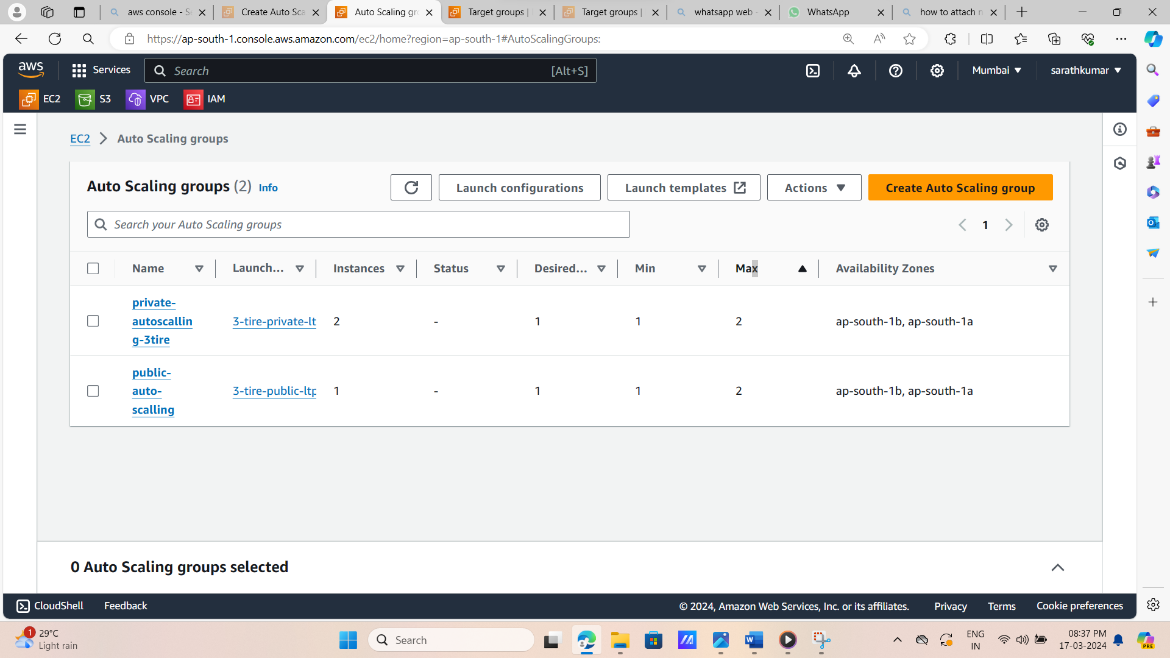
After we create nat gate way,we edit route tables in edit routs in private subnets in diff azs.

5.now create auto scalling group,and assign two public subnets and two private subnets in different az.

Autoscalling process:target grouup,



Autoscalling group



6.now create a rds in two subnets in 2 diff az.

Now create a data base subnet group, and create rds in there subnets with any one engine .its rds connect on only vpc connection.

