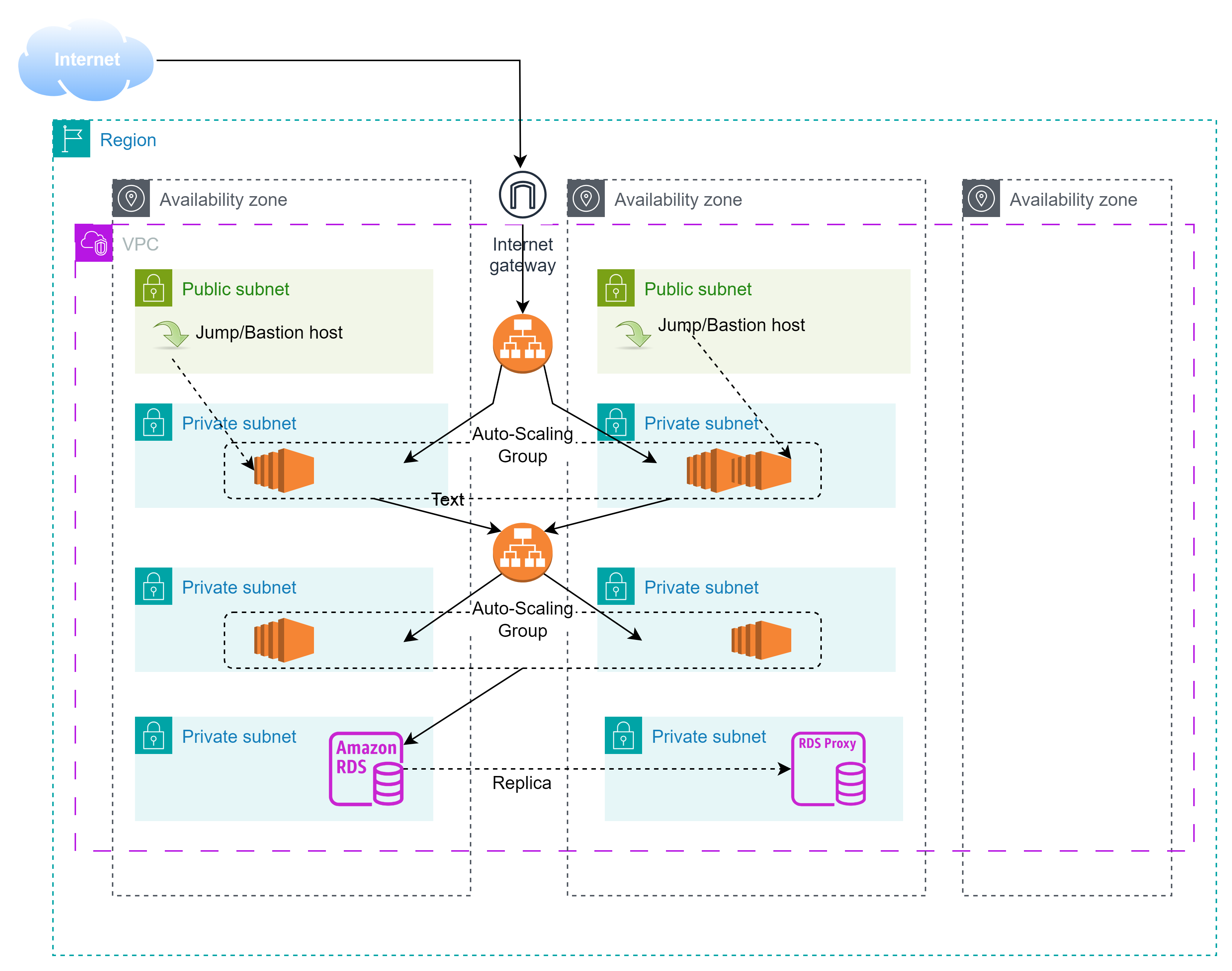
AWS 4 tier architecture with autoscaling

1)Jump/Bastion Host

2)Web-Tier

3)App-tier

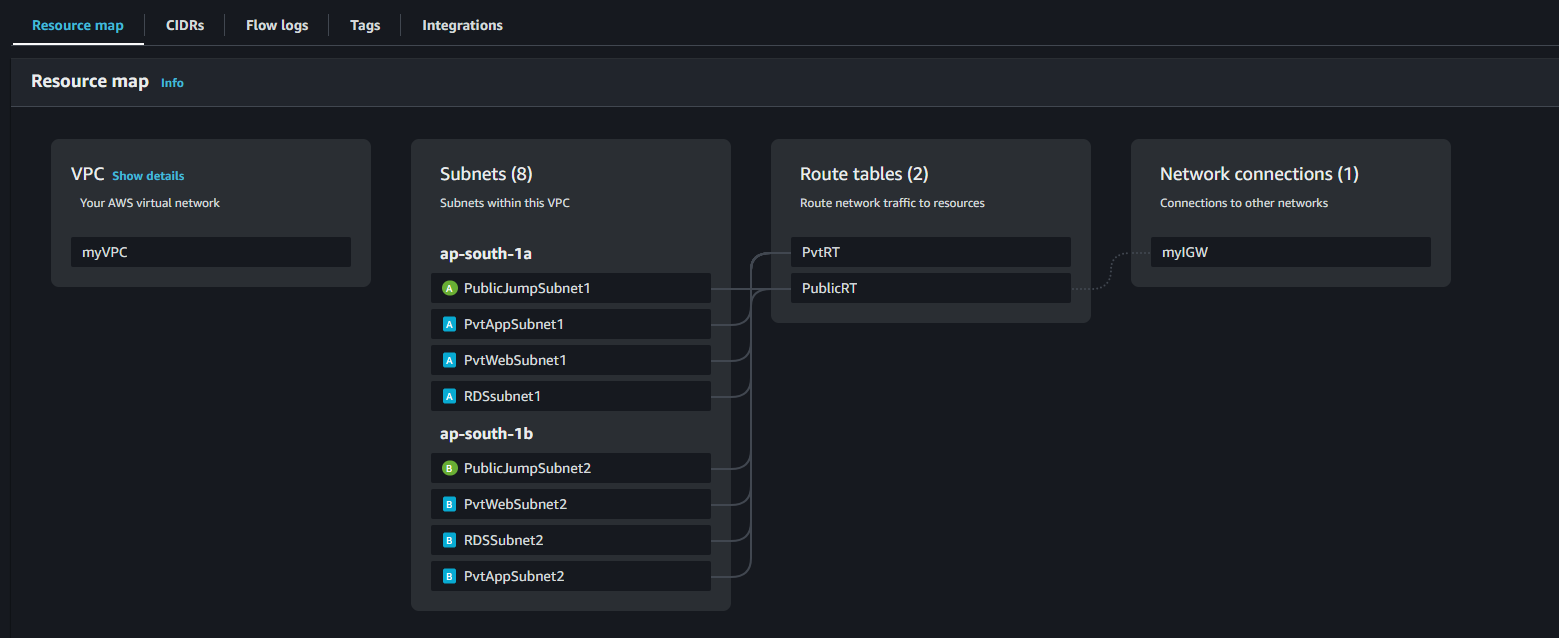
4)Data Base-tier



Here we follow bottom-up approach to resolve dependencies before going up the ladder, The Dependencies in this architecture it Bottom-up so it become easy if you follow bottom up approach

step 1)Create VPC and subnet

Create Custom VPC(Default also can be used) and 8 subnet (We are working in 2 zones only) in which 4 private subnet of presentation and Application tier(2-2 each),2 private subnet for database(RDS),2 public subnet for jump server(Bastion host)



step 2)Create 2 route table public and private and attach private subnet to private route table and public subnet to public route table attach igw to public RT

step 3)Firstly we will create normal 3 tier architecture within public subnet only for easy connectivity and testing(Database in private)

DATABASE

a)Creating database-first create database subnet group - select Private DB subnets in different Availability zones within your VPC

b)Then go to create database section select method as standard create and engine type as MySQL(In templates Free-tier)

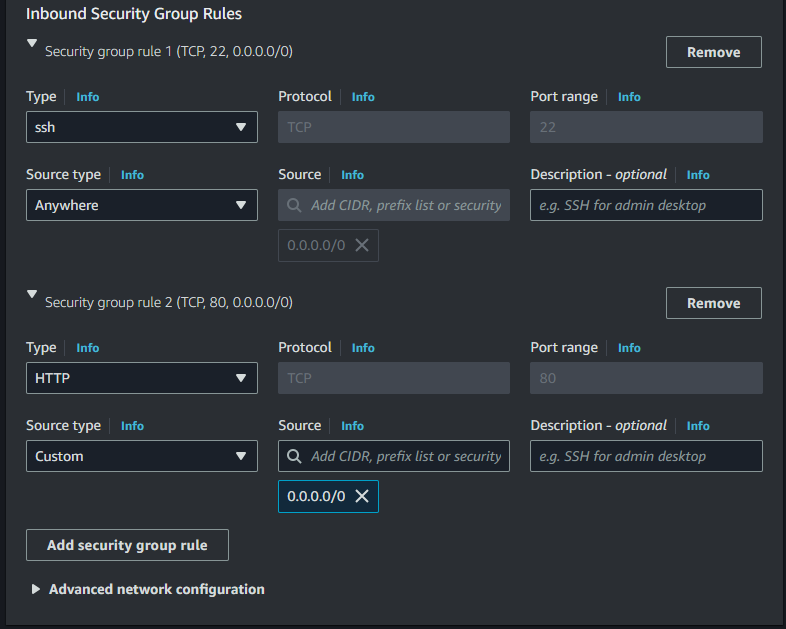
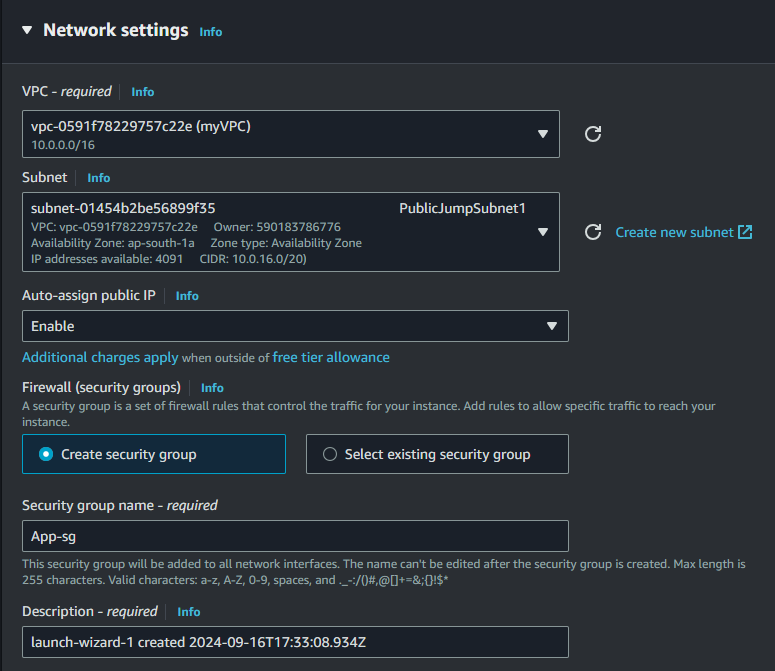
Give database any name and carefully give master-username and remember it (Here I am giving it as root) choose self-managed for credentials and give master password

Then directly go to connectivity choose Don't connect to ec2 compute resource and choose your VPC and DB subnet group we created before (3.a)

choose public access as NO choose any Security Group just it name that you have to remember(I given DB-sg) choose Availability zone if you needed I chose no preference and just create database

APP TIER

a)choose launch instance ,OS Amazon-Linux, type t2.micro(free),your key pair login and select following setting and launch instance

Now connect to instance and follow further steps

Use command

1) sudo yum install nginx php-fpm –y (To install php and nginx)

2)enable both of them for AMI using command

sudo systemctl enable nginx php-fpm

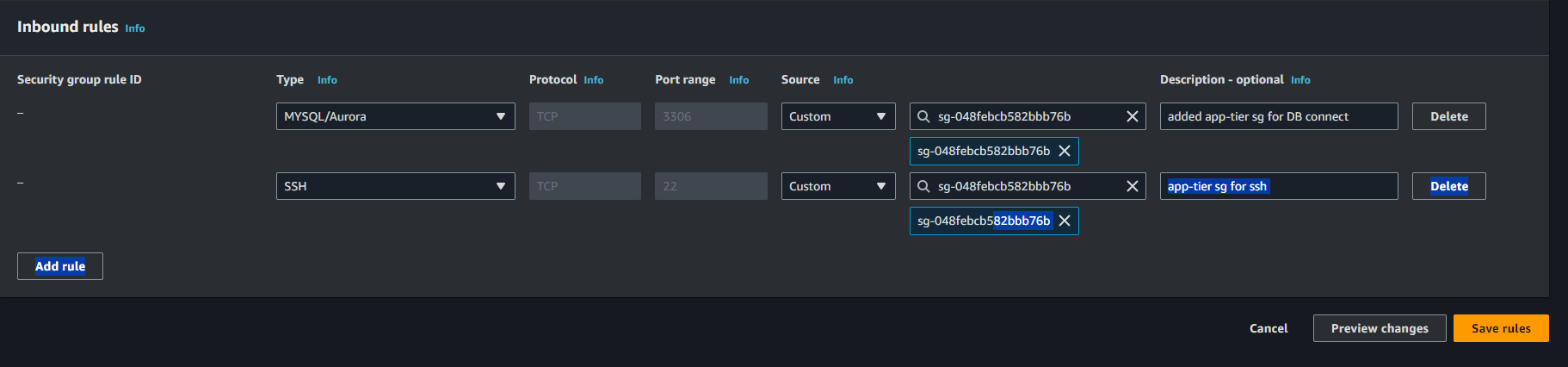
3)Install mysql for connection to RDS database

sudo yum install mariadb105 –y

sudo yum install php8.3-mysqlnd

4) Go to security groups and in database security group inbound rules add app tier security group for mysql –aurora port 3306

And also add ssh for app tier

And now connect to database using command

sudo mysql –u root –p –h <Database-endpoint- url>

And then enter password you got connected to RDS database exit for now by typing exit

Now make sample php file in /usr/share/nginx/html

And make file using - sudo nano sample.php

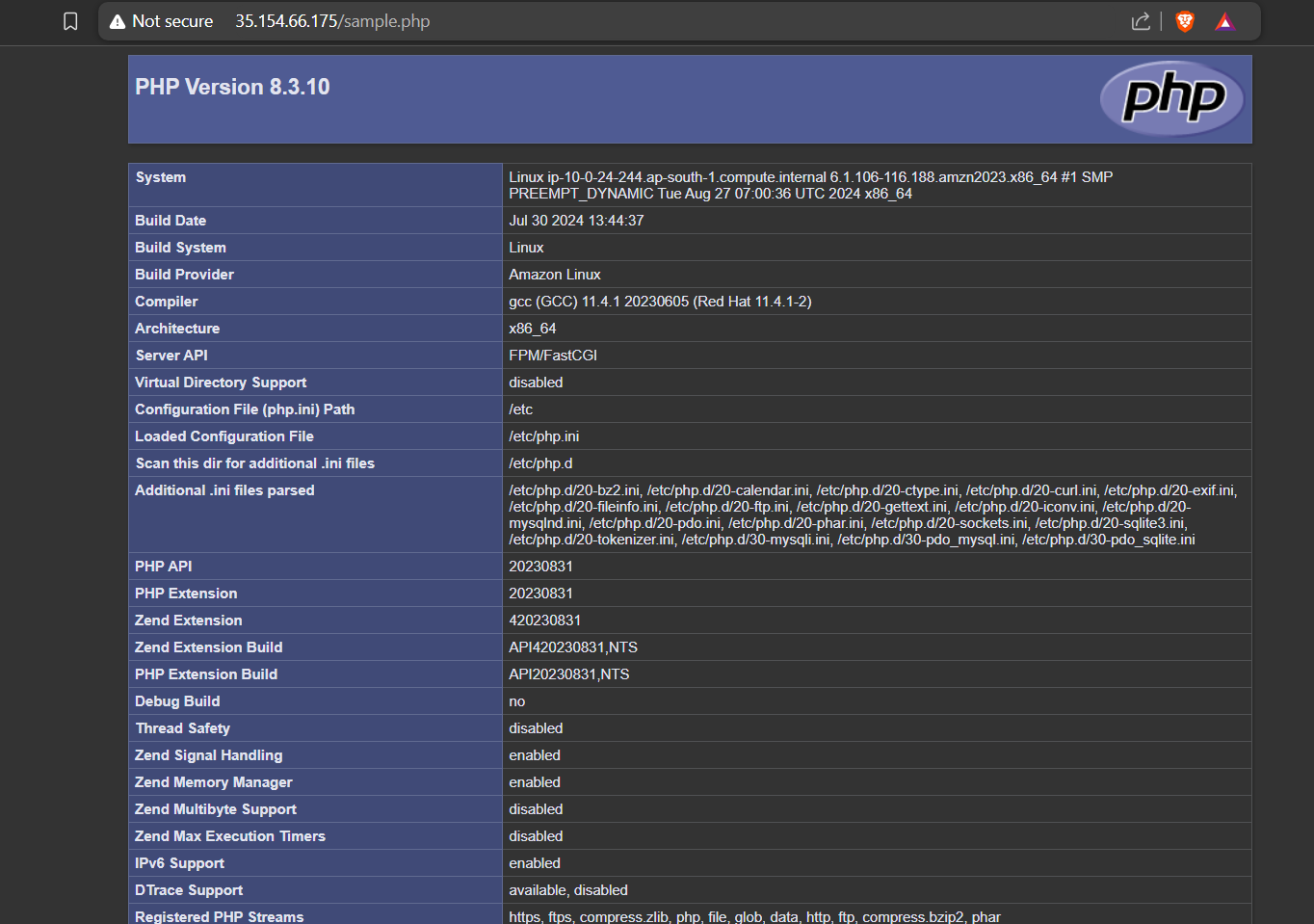
<?php

phpinfo();

?>

Save file and reload nginx using- sudo systemctl reload nginx(if not started – sudo systemctl start nginx) and then copy paste your app server public ip/sample.php

You should see:



Then create another file submit.php(code in github repo) in /usr/share/nginx/html

Now connect to your database using – sudo mysql –u root –p –h <Rds endpoint>

And type this commnads that creates db and table in rds

create database facebook;

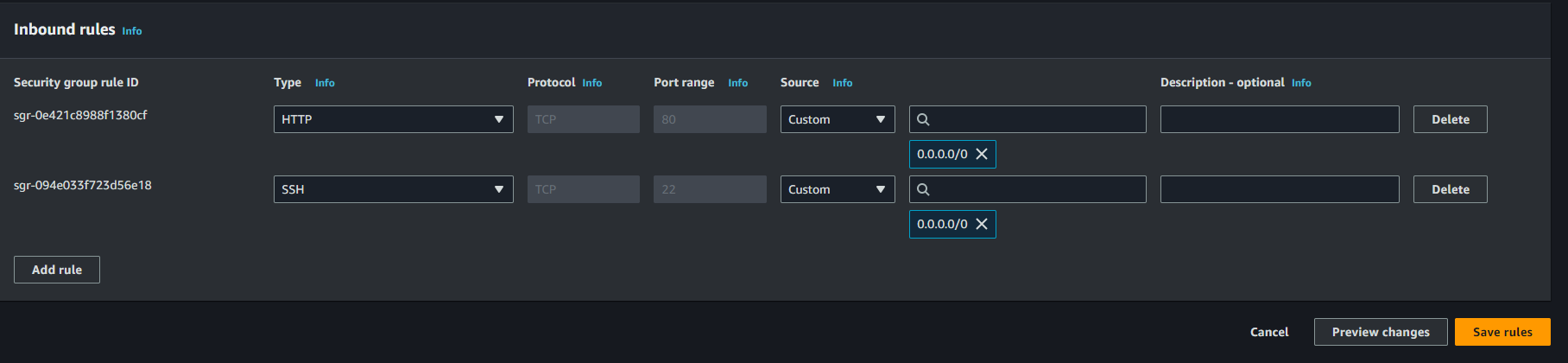
use facebook;

create table users(name varchar(50),email varchar(50) unique,password varchar(30));

Exit

WEB TIER

1)Launch web tier instance in public subnet only and use security grp with following rules (give name as web-sg)

And connect to web instance

And install and enable nginx using-

sudo yum install nginx –y

sudo systemctl start nginx

sudo systemctl enable nginx

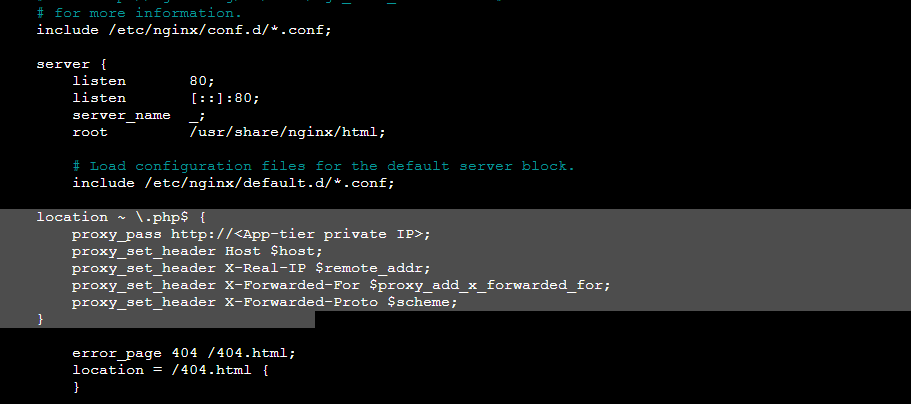
And then go to /usr/share/nginx/html/ and create file forms.html (in github repo)

And reload nginx and see if working or not using ec2 publicIP/forms.html

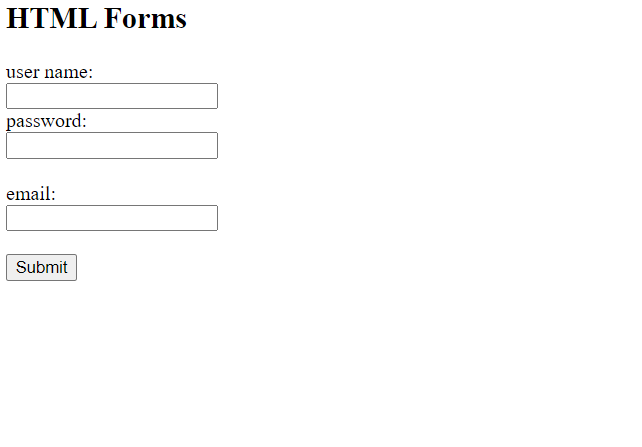
If its working move further

Go to nginx config file using cd /etc/nginx

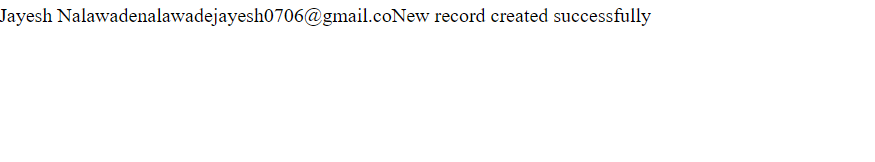
Sudo nano nginx.config and type following highlighted part at same space change App tier app by your app tier priavte ip

Now reload nginx using – sudo systemctl reload nginx

And again type publicIp/forms.html and fill your details

And you’ll see this screen after filling details

which shows your details for confirmation purpose(Ik its not good website but I made it for just knowledge purpose and you can add your content/website accordingly )

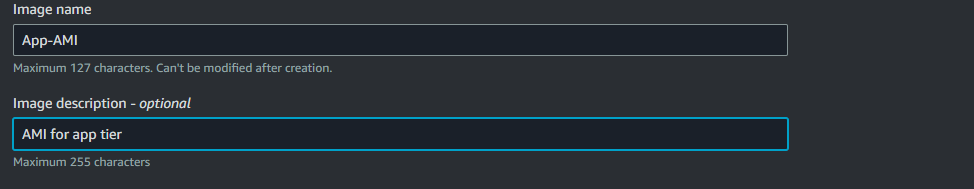
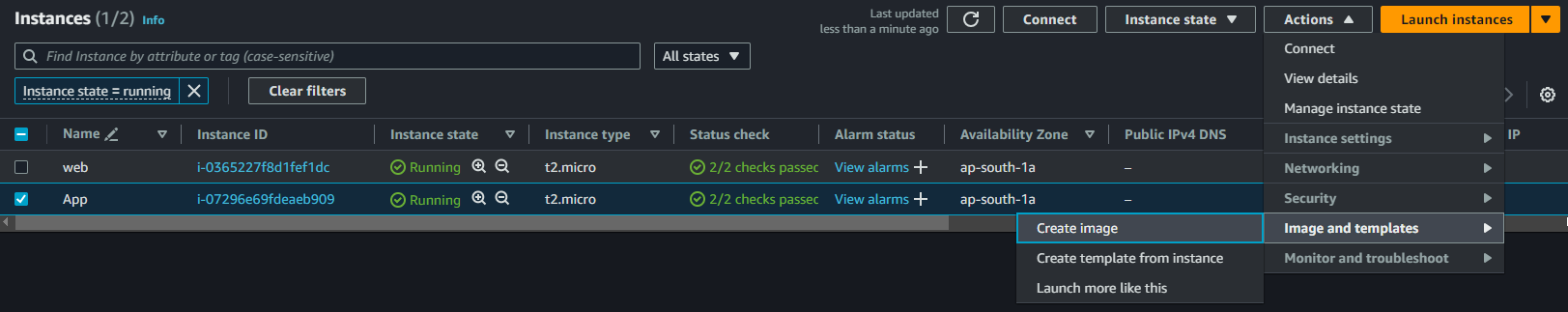


Now your 4 tier application base is completed we can move further now

Step 4)Creating Autoscaling-Group for app tier

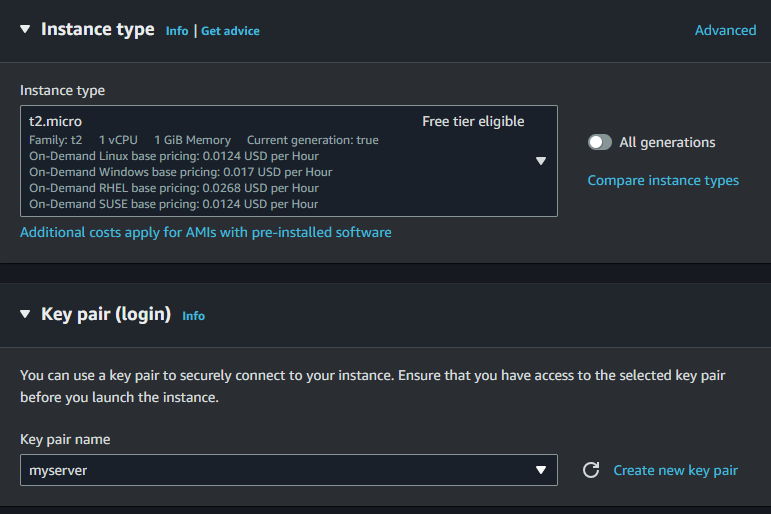
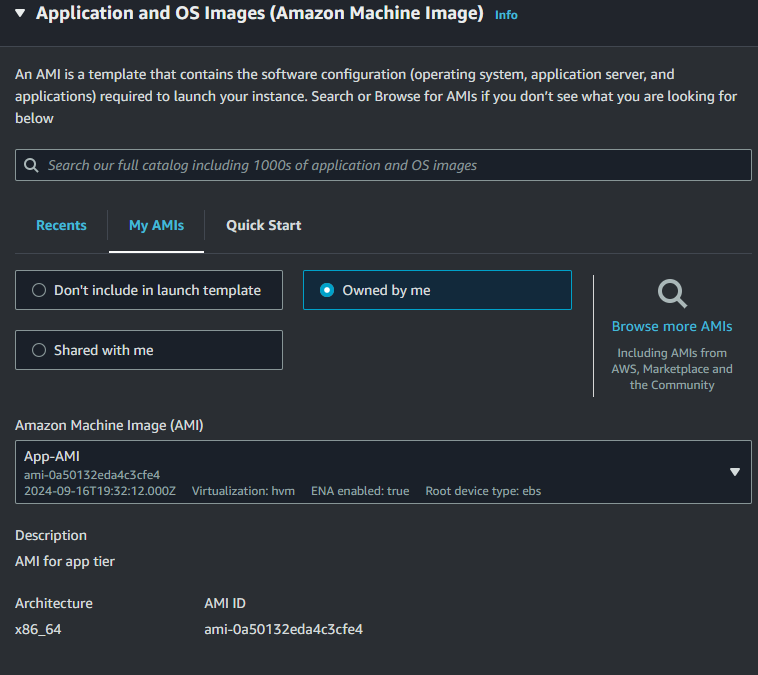
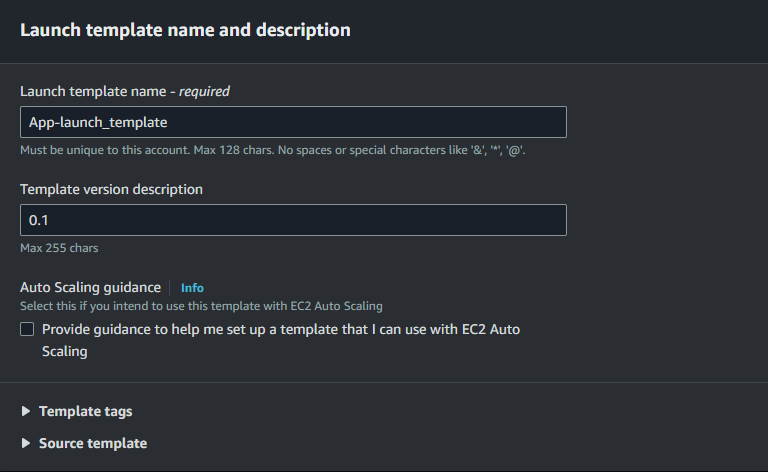
Our App tier instance is sorted and its request go to rds so we can create Auto-sacling-group now

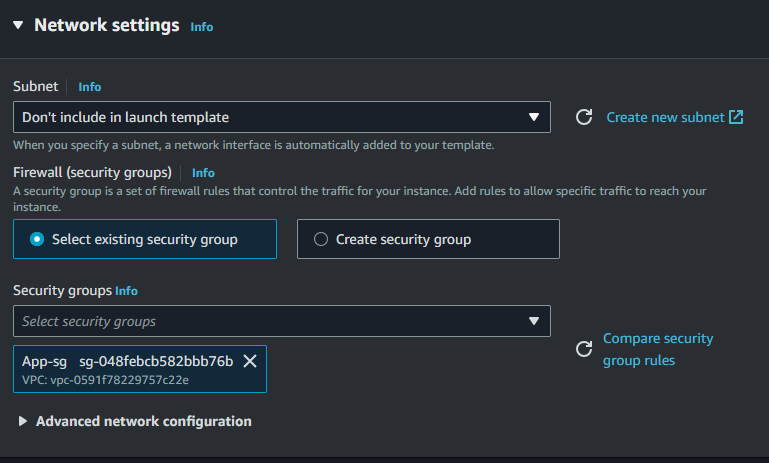
a)first create ami of our App instance

and create AMI

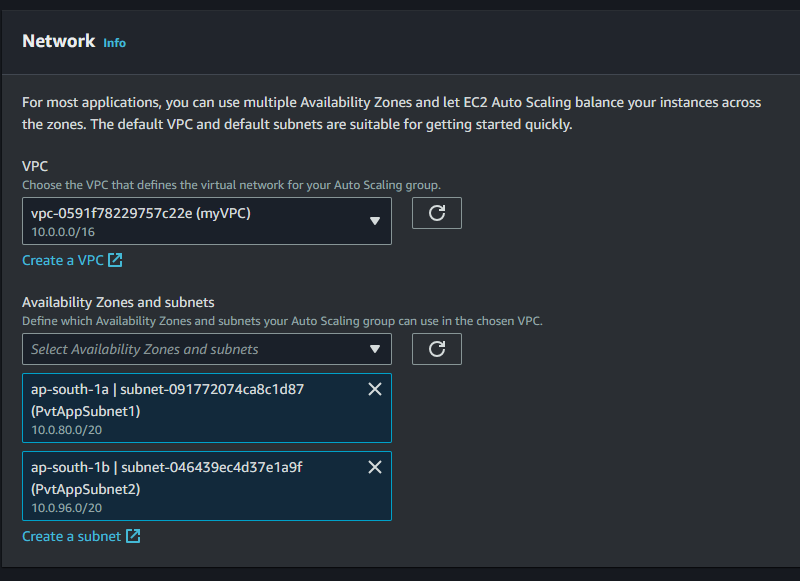
Now create launch template before going to Auto scaling group

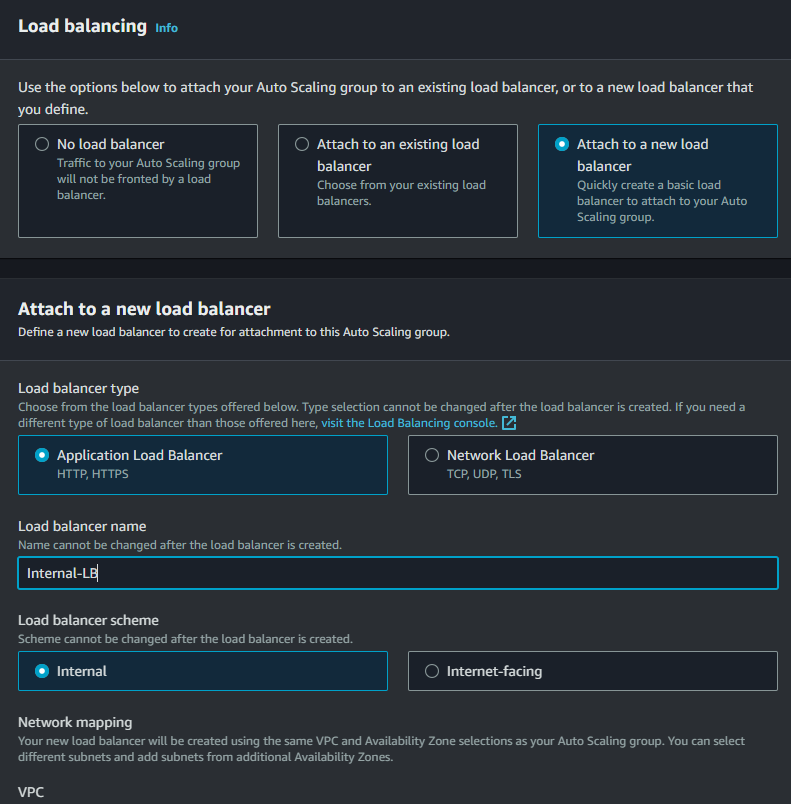
Launch template

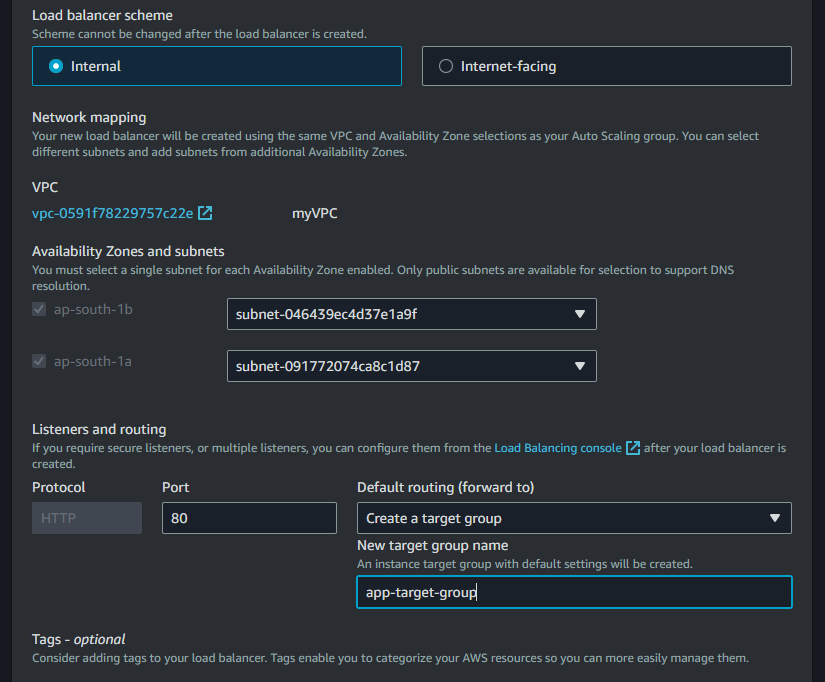
use your key pair here

And now just launch template

Now go to Auto scaling group give name and select app-launch-template

this AZ and subnet are where your instances through auto scaling will get launch And now click next

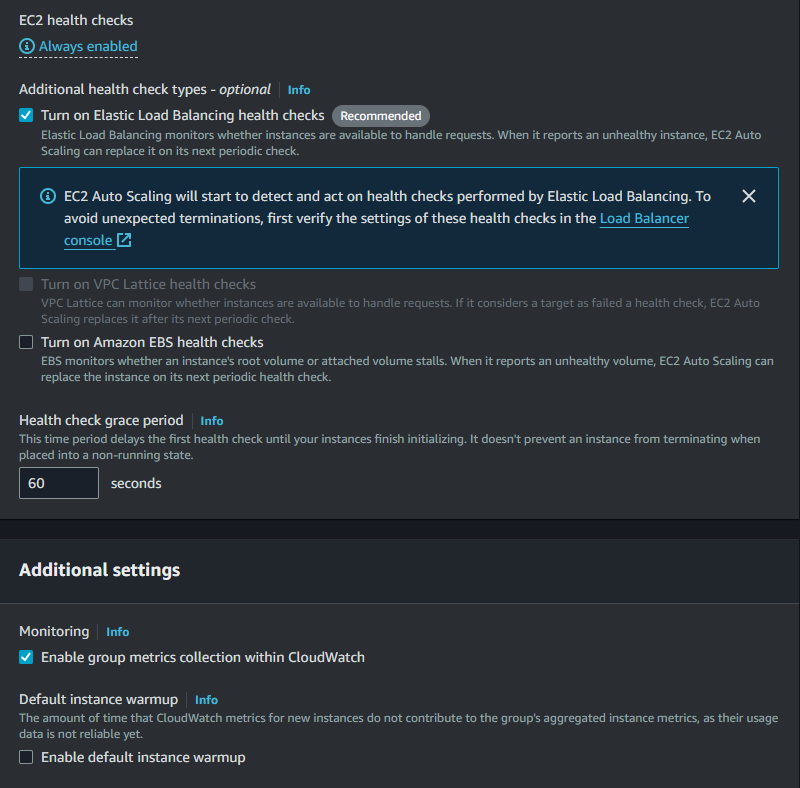


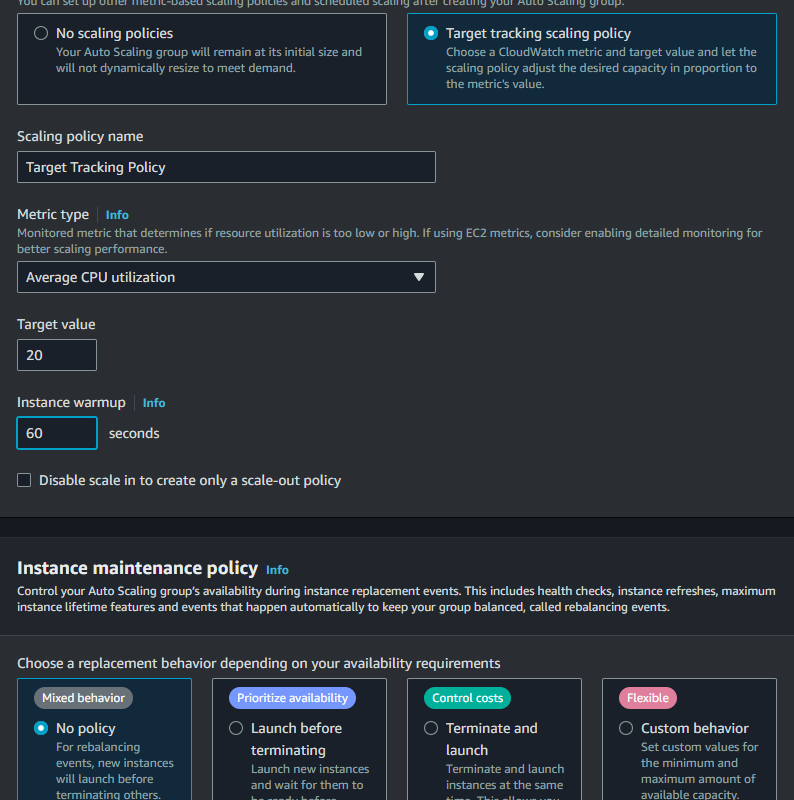
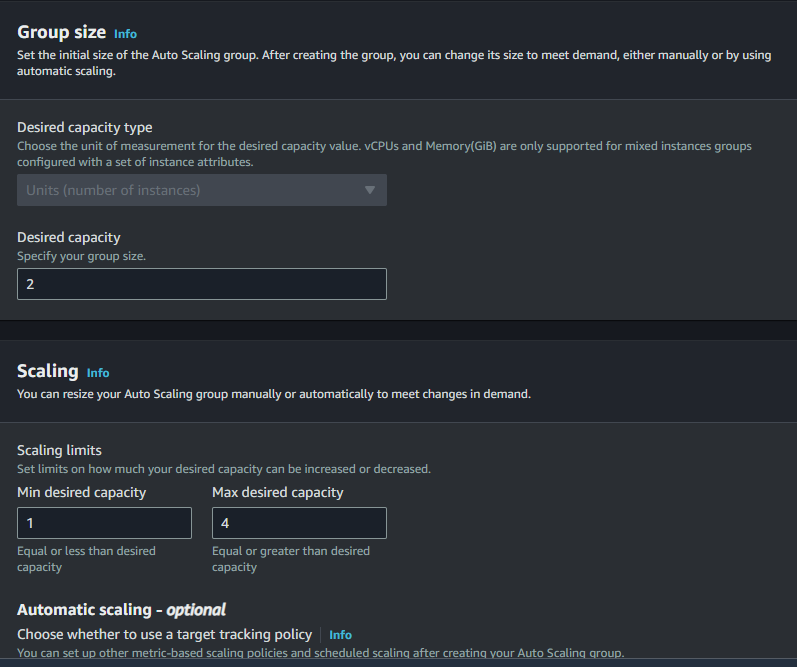


This subnets are where your load balancer will launch

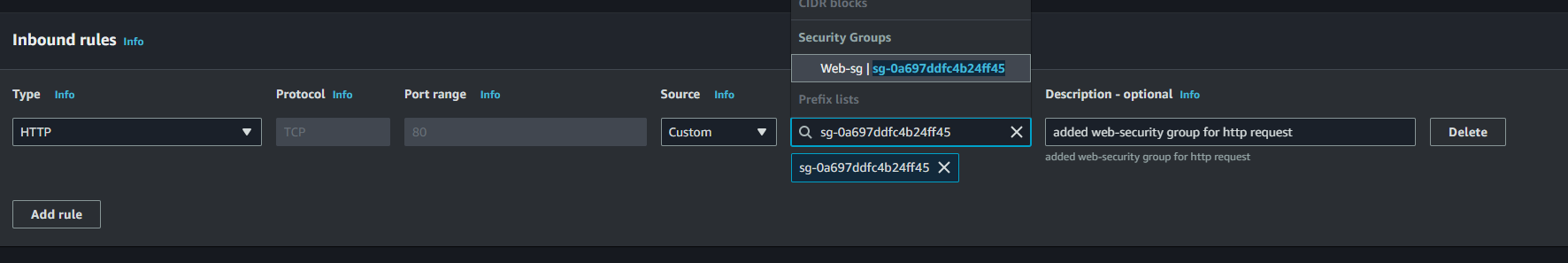
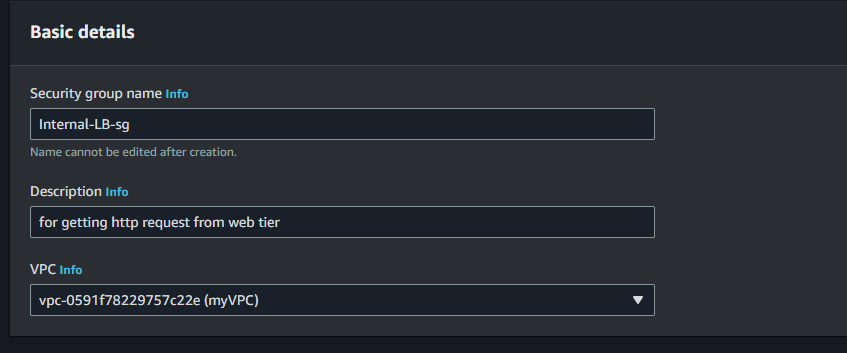
Select same subnet as app subnet cause both instances and load balancer should be in private subnet (We can also select pvtWebSubnets here)

Select create target group and name anything

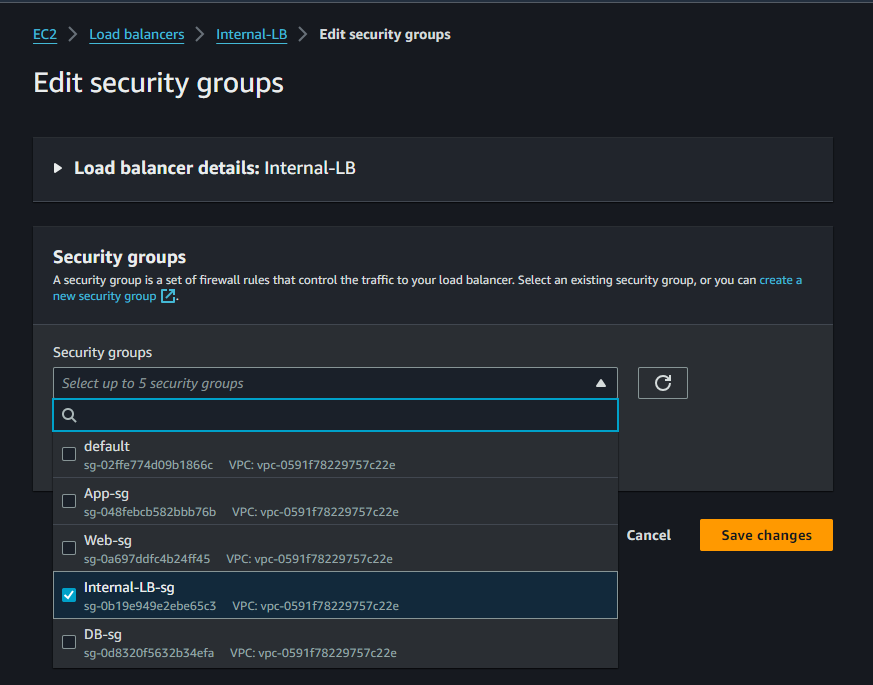
Do same as I did here and click next

select same as I did No Policy and click next add sns topics and tags if you need and just create auto scaling group

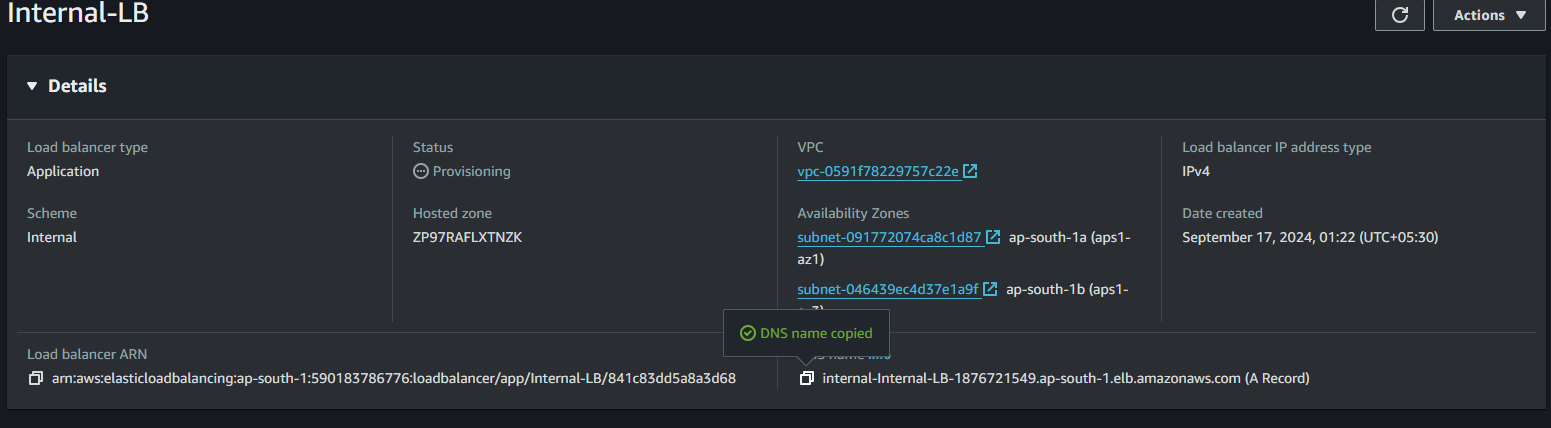
Now go to security group and create security group for internal (app)-loadbalancer

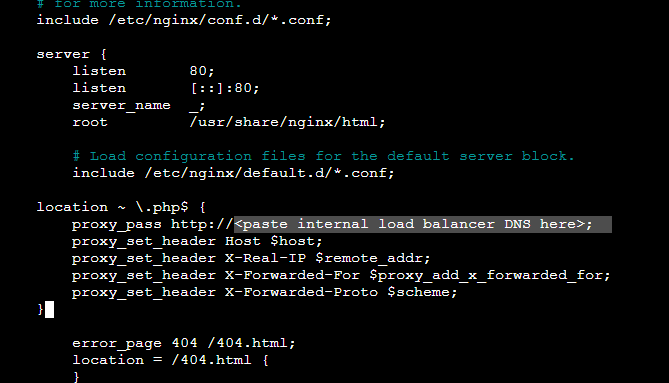
and create sec grp

Now go to internal load balancer and in its security change(edit) the security group to new security group we created



Now copy internal-load-balancer DNS name

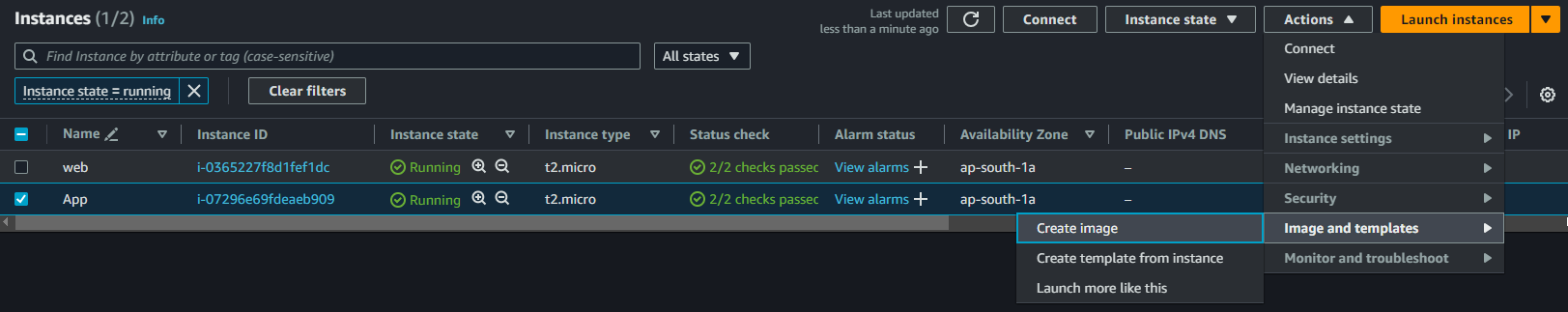
And now connect to our web instance and do sudo nano /etc/nginx/nginx.conf

Paste your internal load balancer DNS and save it

Again reload your nginx using – sudo systemctl reload nginx

Step 5)Creating Auto scaling Group for web tier

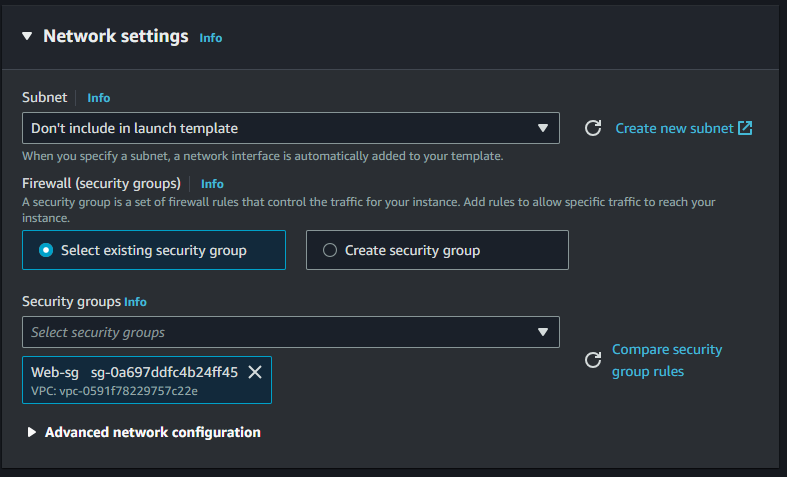
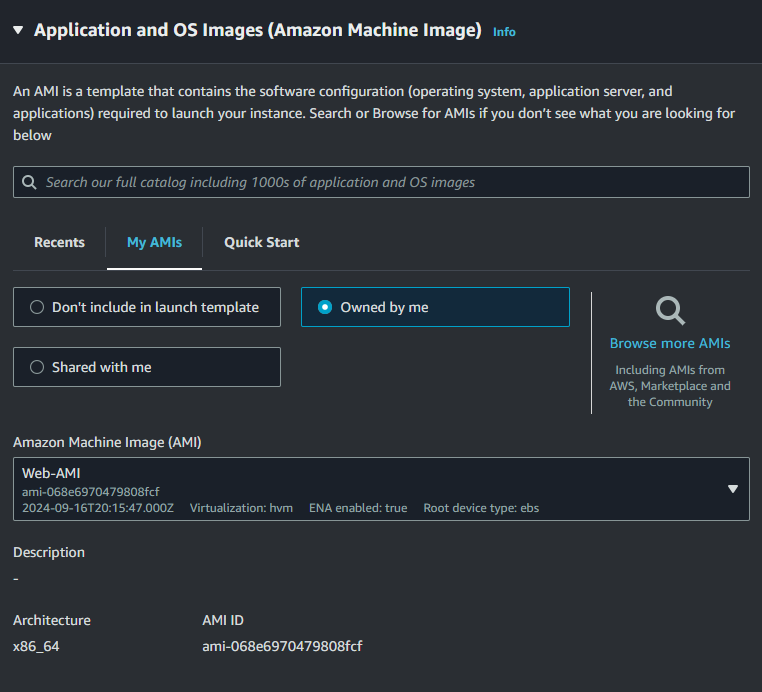
a)first create ami of our Web instance (Similarly we have done for app instace)

and create AMI

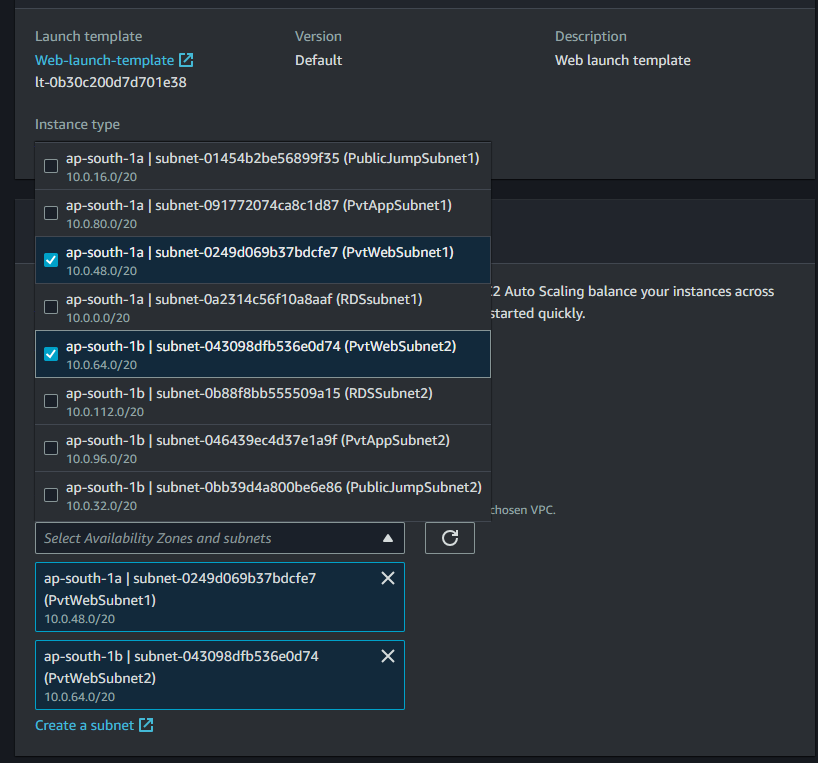
Now create launch template before going to Auto scaling group

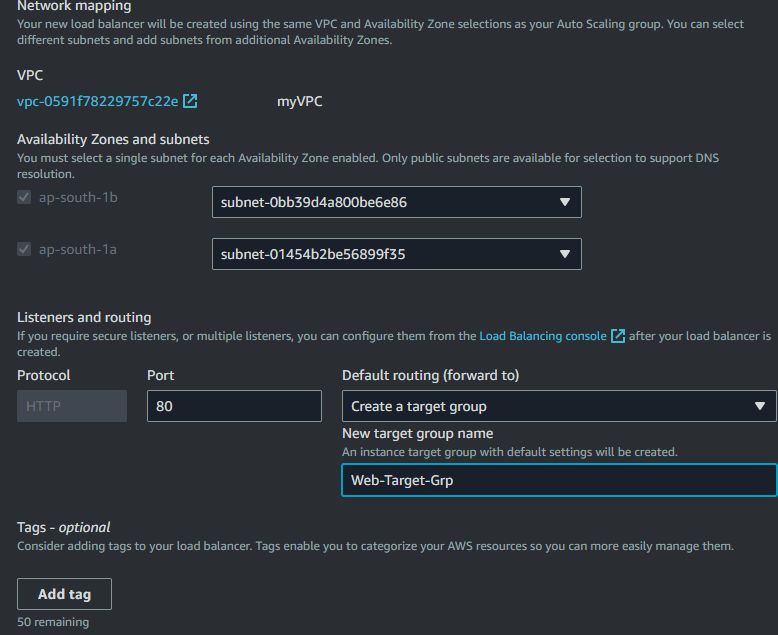
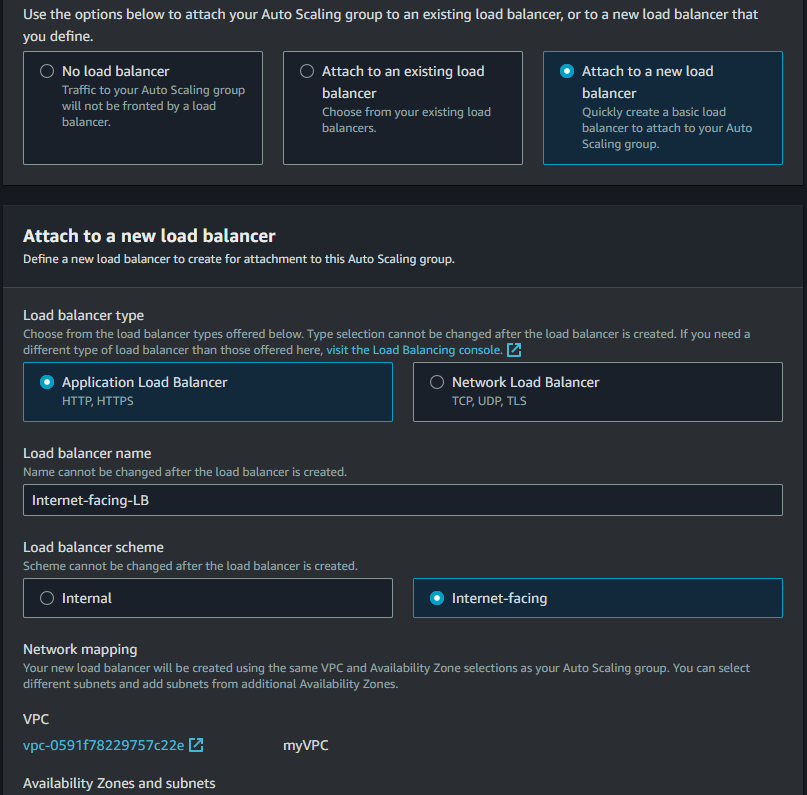
Launch template

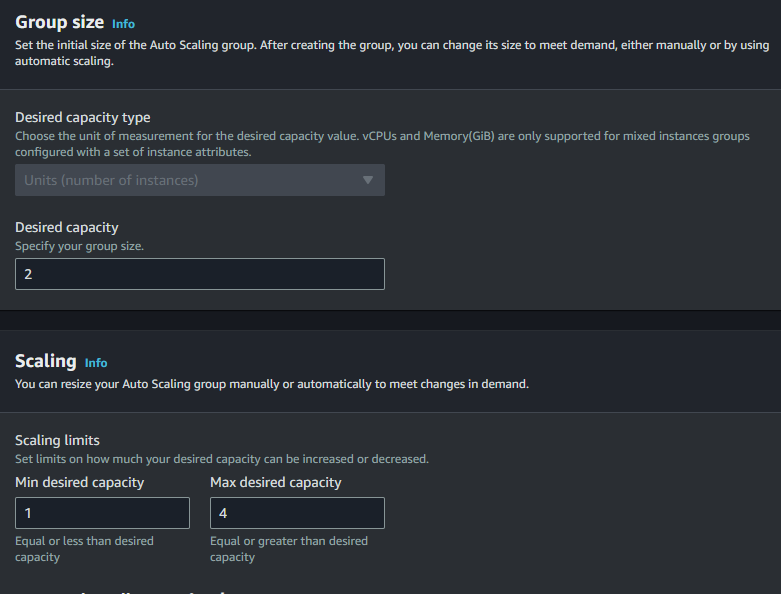
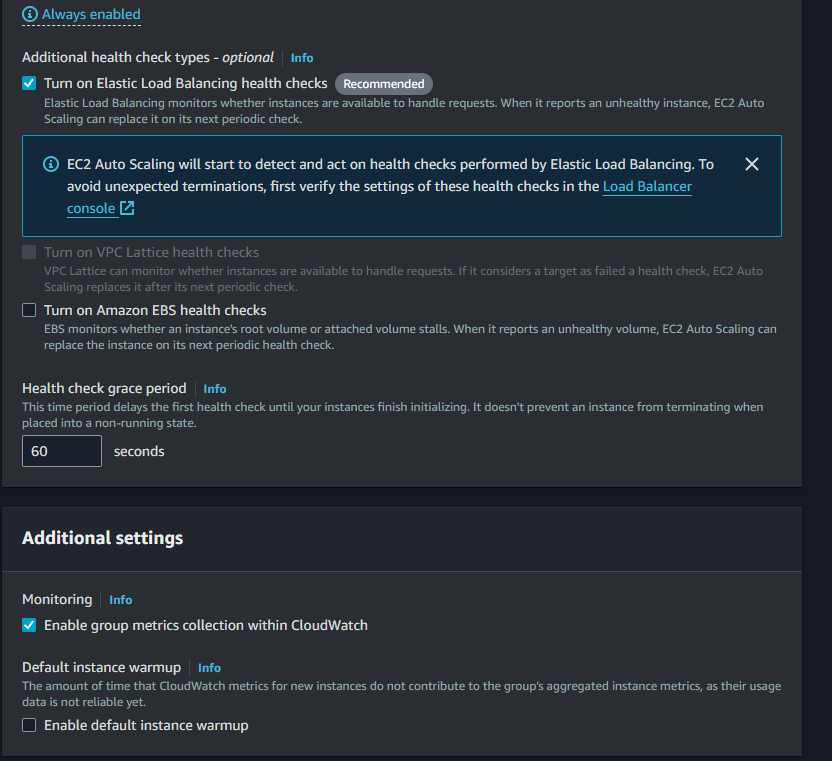
Create launch template similarly we created app launch template just select web server related data such as:

Don not forgot to select t2 micro for instance type and now create launch template

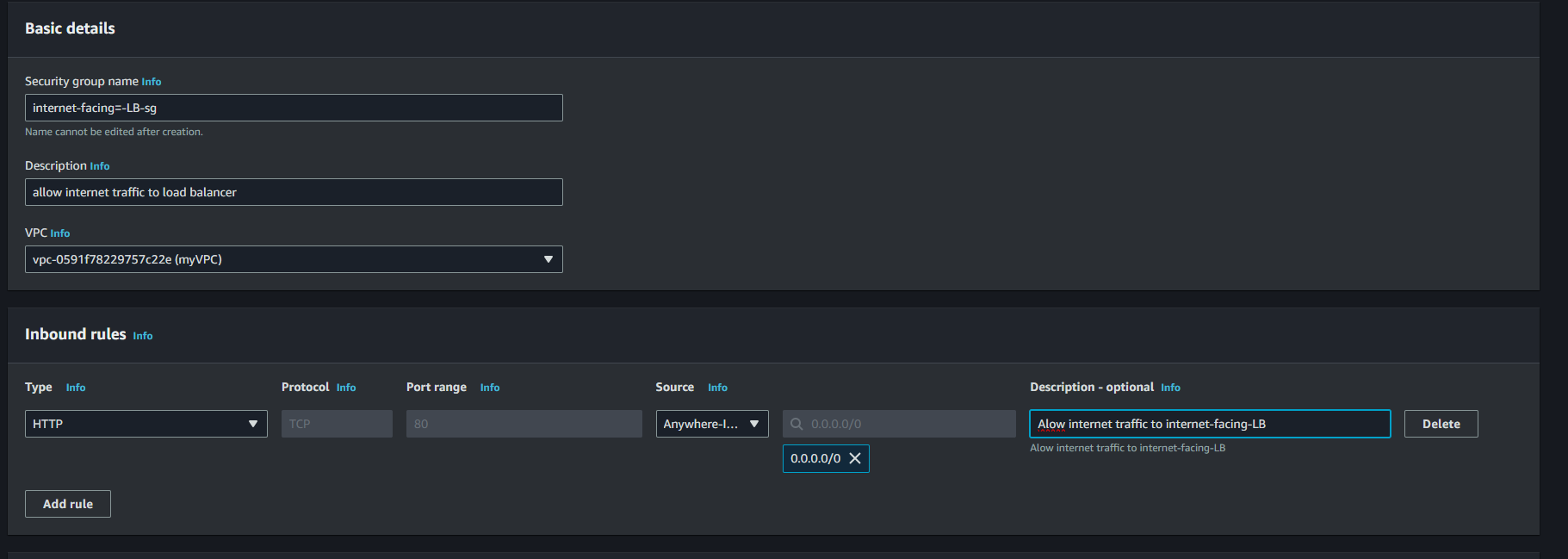
Now go to Auto scaling group give name and select web-launch-template

Select private subnet created for ec2 web instances get launch in this subnets in autoscaling

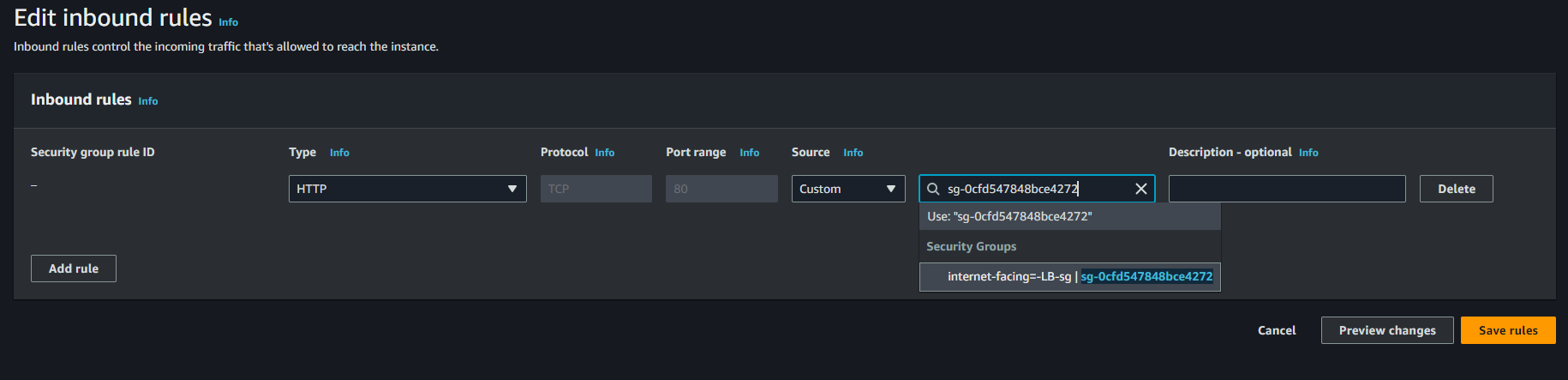
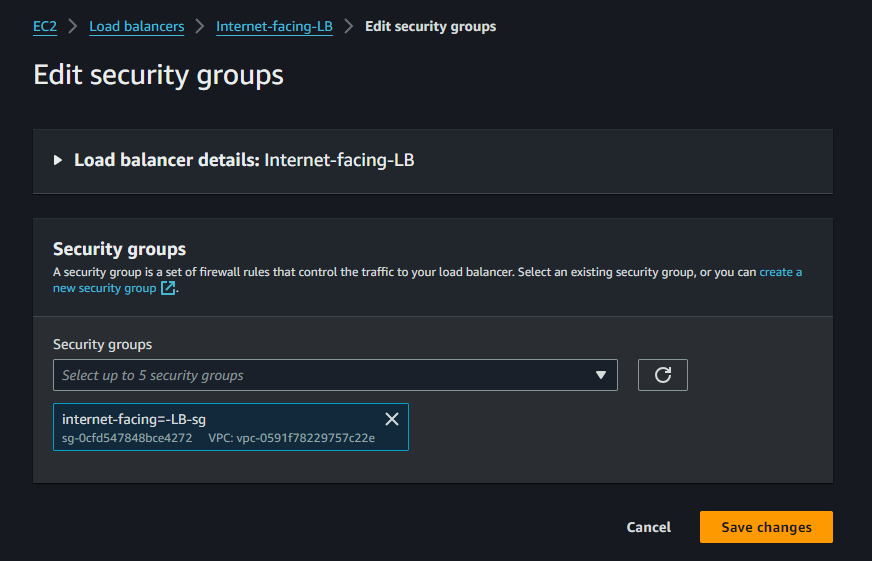
select public subnet here cause your LB gets the request from internet so it should be in public subnet

Just select option as I did and click next and select SNS and tag if you need and just create auto-scaling-group

Now go to security group and create security group for internet-facing (web)-loadbalancer

Create security group and copy its(internet-facing-LB-sg) security id

Now go to web-sg(Security group ) and do the following changes

Change the security group of internet-facing-load balancer to its own security grouNow our structure is completed still check the security groups inbound rule as:

DB-sg > should have MySQL/Aurora from App-sg

App-sg > should have HTTP from Internal-LB-sg

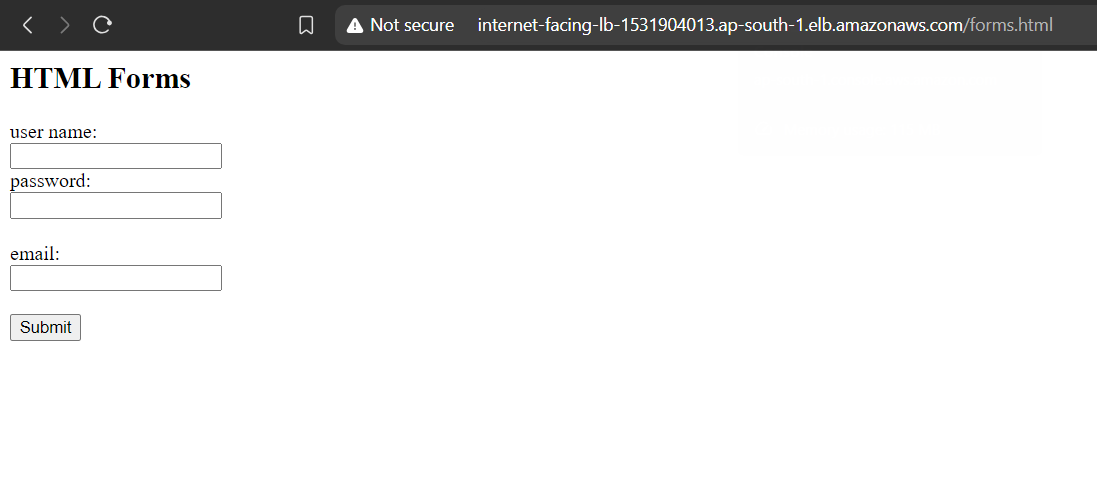
Internal-LB-sg > should have HTTP from Web-sg

Web-sg > should have HTTP from Internet-facing-LB-sg

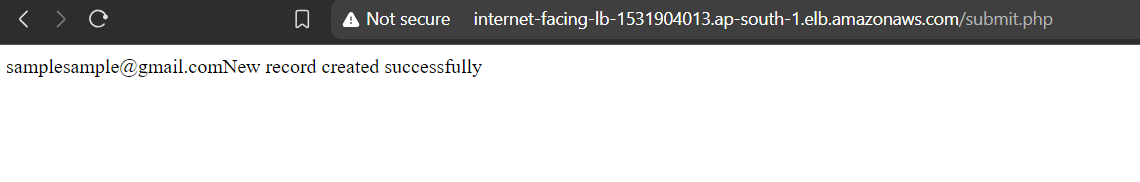
Internet-facing-LB-sg > should have HTTP from 0.0.0.0/0

Check it carefully most of the errors occur in security froup only if you followed all other steps carefully

Now take DNS of Internet facing LB and paste it with /forms.html

You should get this page

After filling details and submit

If you have seen this screen then you have created 4-tier architecture with Auto-scaling successfully, Congratulations

Now Close and delete all the services you have started while creating to avoid huge amount of bill, Thanks !

Source github link - https://github.com/Jayesh0706/Devops/new/main/4%20tier%20architecture