**AWS Overview**

**Assignment/ Questions**

https://docs.google.com/document/d/1C2JLcZCaYve9ftLnvwdM7g5H1KJGDKPMEfH-vO75NJE/edit?tab=t.0

*Q1)List out the types of instance base on the pricing model and write a brief about your understanding about it.*

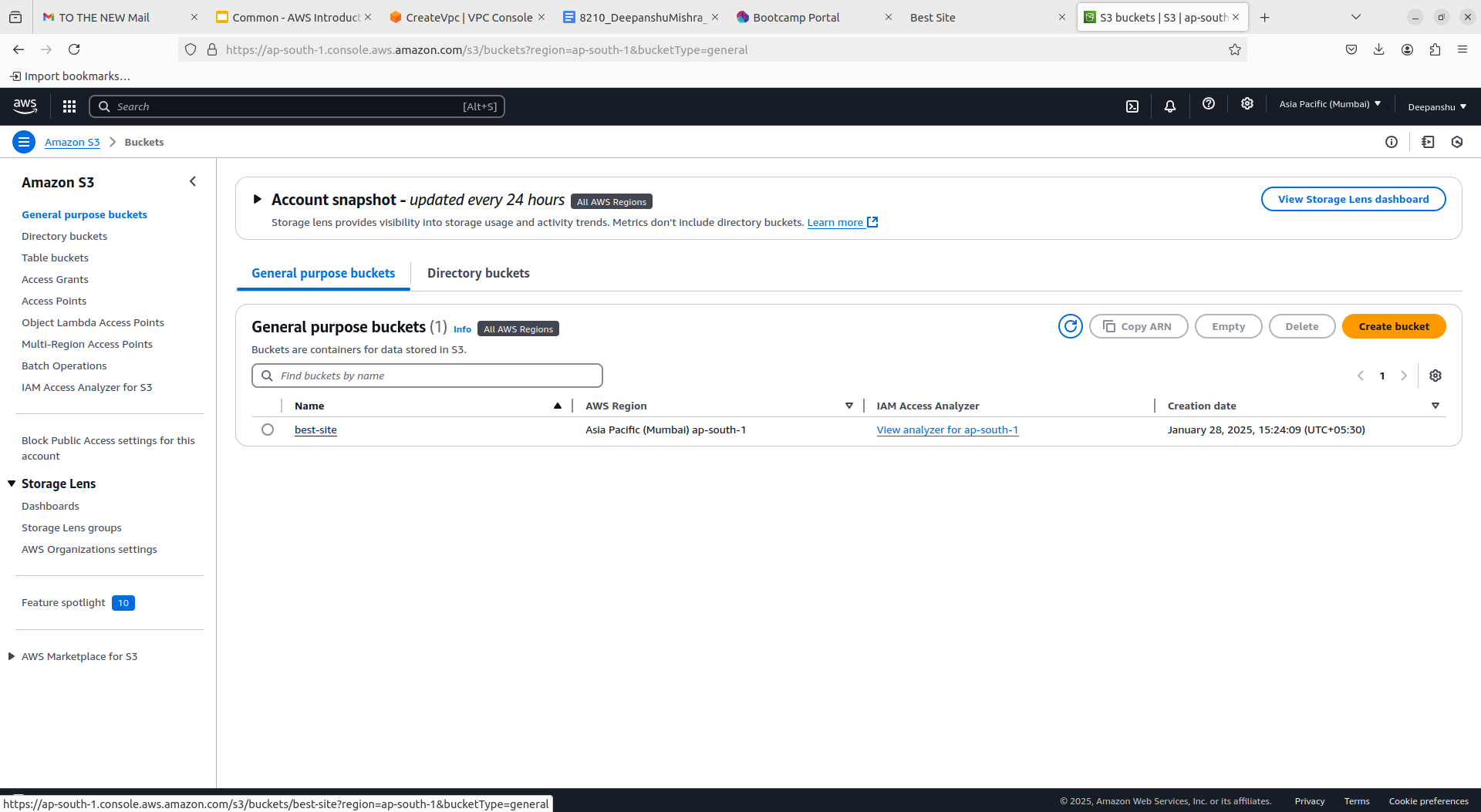
A1) The AWS pricing model followse a pay-as-you-go approach where you only pay for the resources you actually use. This way is flexible and cost-effective and allows orgs to scale their infrastructure based on the demand. No upfront cost is required for most services, you only pay for computing resources you consume, you can terminate resources when they aren’t needed. Volume based discounted pricing is also available as the usage increases. Multiple pricing options to optimize cost based on usage patterns. There is also a free tier available for new users to experiment with services. Billing is granular (per second or per minute for most services). And tools are also available for cost monitoring and optimization.

Types of instances based on the pricing model:-

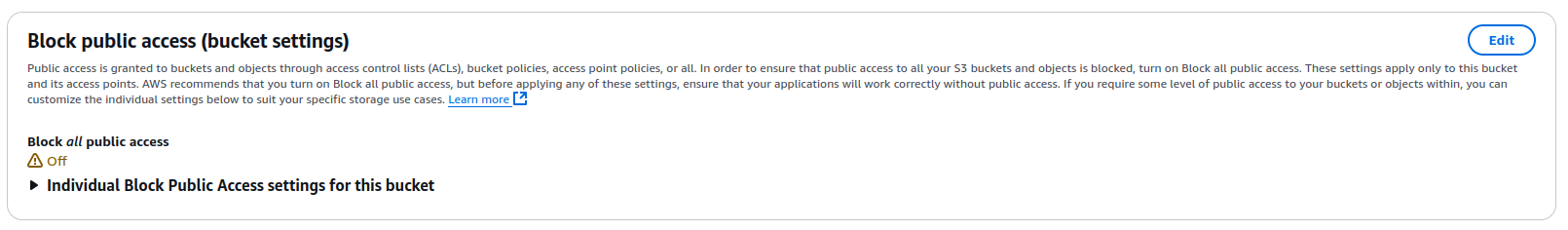
1. **On demand instances -** On-demand instances are flexible and allows the users to pay for the compute capacity by hour or second without any upfront commitment. Its ideal for dev/test environments, short term projects or unpredictable workloads. Comparatively has the highest cost but its the most flexible
2. **Savings Plans -** Offers lower prices in exchange for basically a commitment to a consistent amount of usage ($/hr) for 1 or 3 years. This option is flexible across instance types, regions, OS and AWS services providing cost savings without actually sacrificing agility. Available in 2 types - compute(applies to ec2, lambda, fargate. Most flexible) and EC2 instance savings (specific to only ec2 but offers higher discounts).
3. **Spot instances -** Model where you utilize spare AWS capacity at discounted rates (~90%) without upfront commitment. Ideal for batch processing data analytics, optional tasks and best for fault tolerant and flexible workload.
4. **Reserved Instances -**  Save upto 75% cost compared to on-demand by committing to a specific instance type in a specific region for 1 or 3 yesr. Ideal for a stable and predictable workload. Available in 3 types - standard(steady-state usage, most cost effective), convertible (flexibly change instance attributes in the future) and scheduled(reserved for specific fixed times)

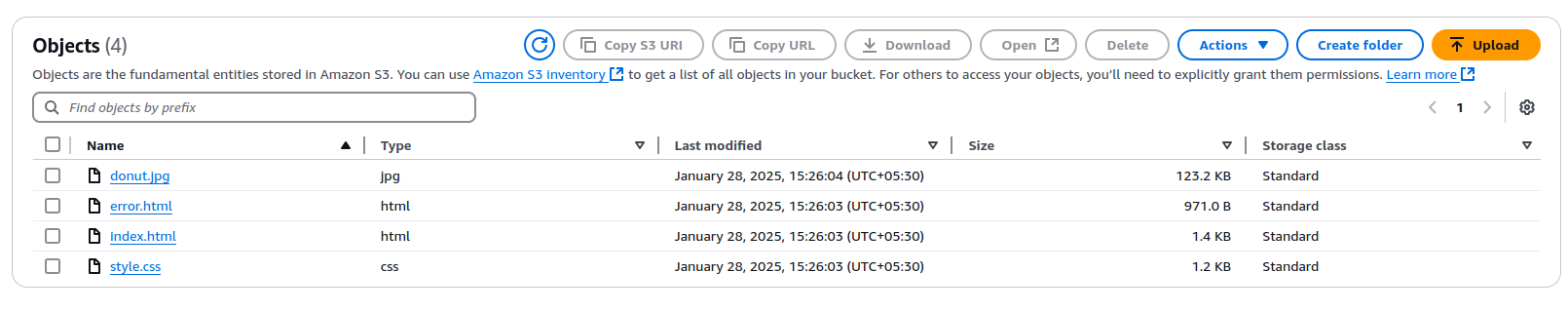
*Q2)Host a static website in S3.*

Created an s3 bucket called best-site



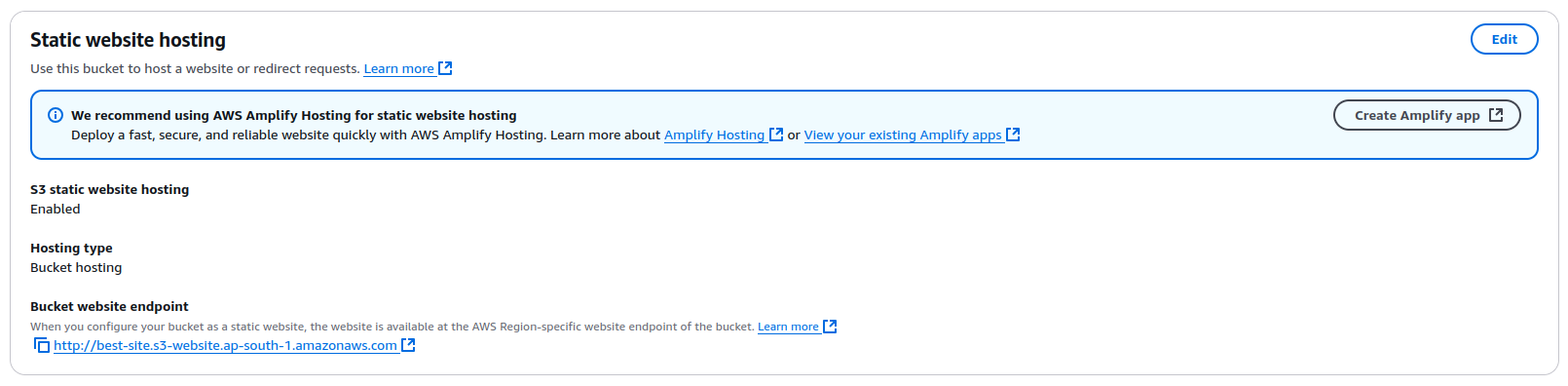
Turned off access blocking such that the site can be accessed from anywhere



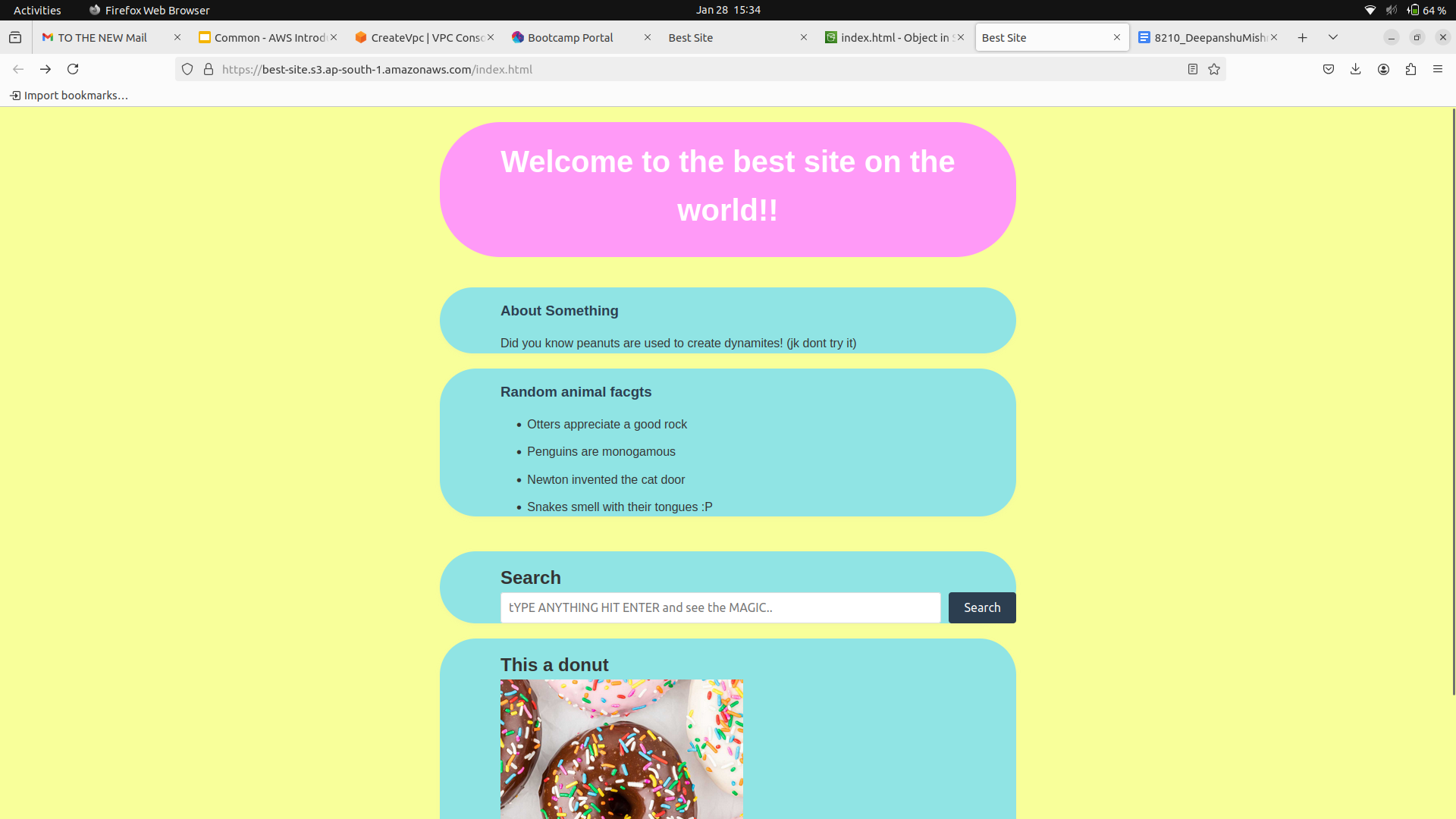
Upload website content in the bucket

Similarly for every object give everyone read access for viewing the site



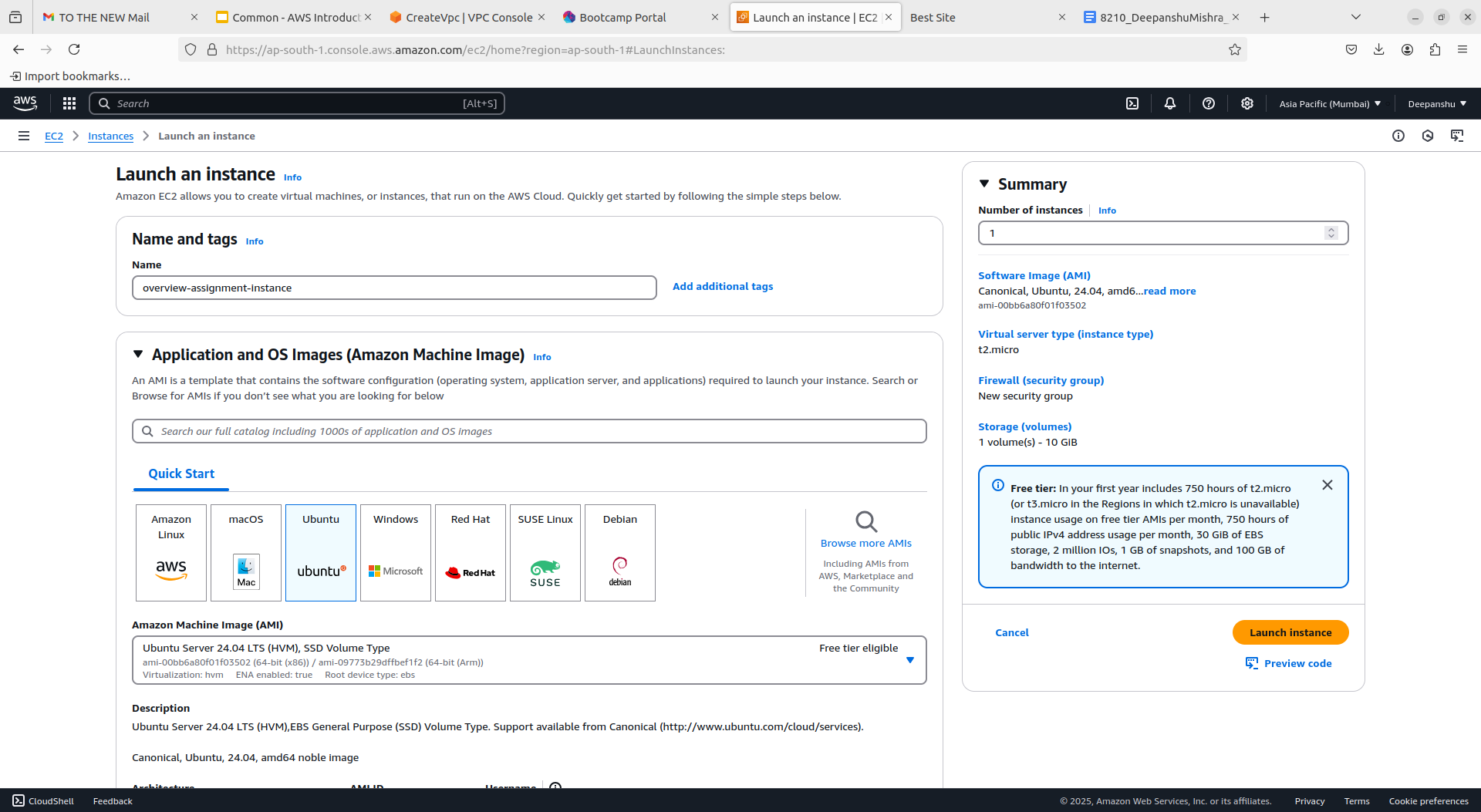
Turn on static website hosting in bucket properties

The site is now hosted! (https://best-site.s3.ap-south-1.amazonaws.com/index.html)

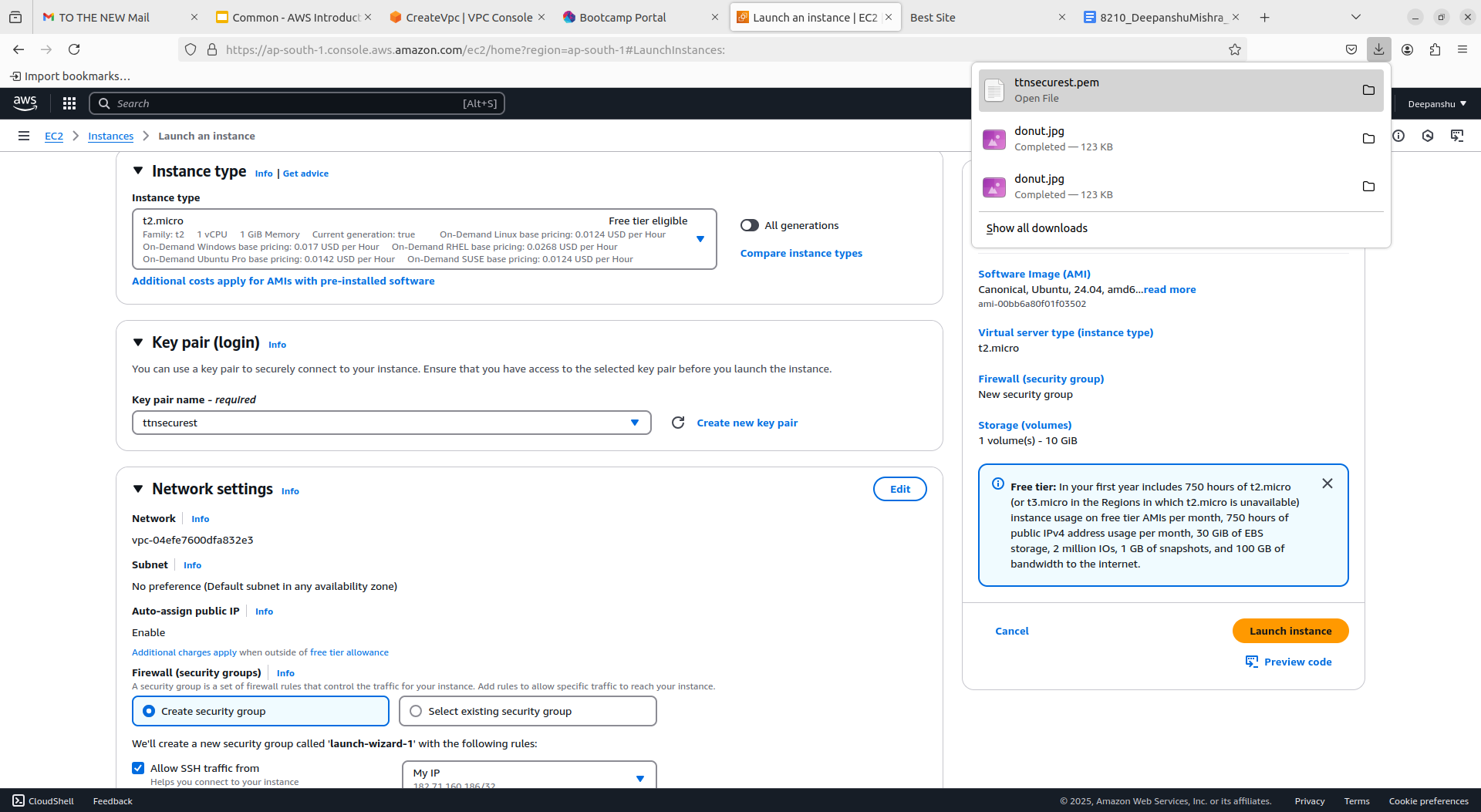


*Q3)Launch an Ubuntu EC2 instance on AWS, with 10GB root volume, and SSH from your local machine using the private key.*

From AWS dashboard click on EC2 launch instance, select Ubuntu as AMI



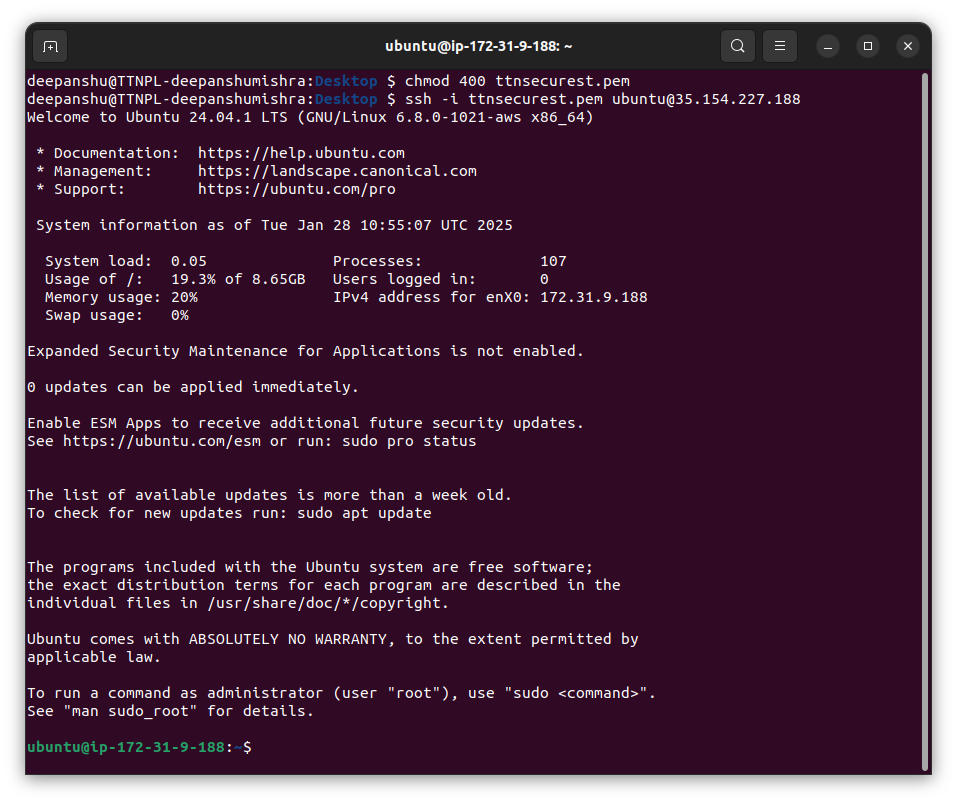
Select the instance type as t2 micro, click generate key-pair and move the downloaded file to an accessible location.



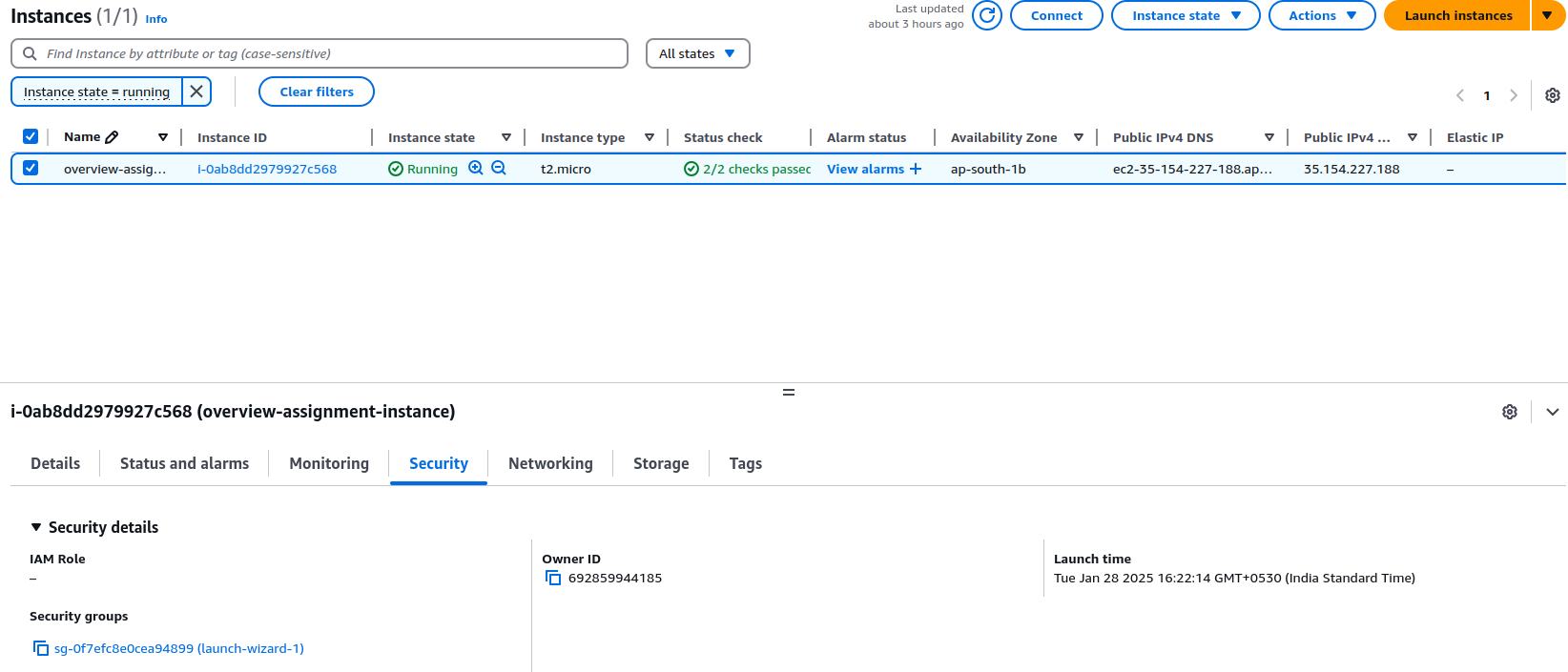
Select My IP from the dropdown next to “allow ssh traffic from” and put in 10 in configure storage, click on launch instance



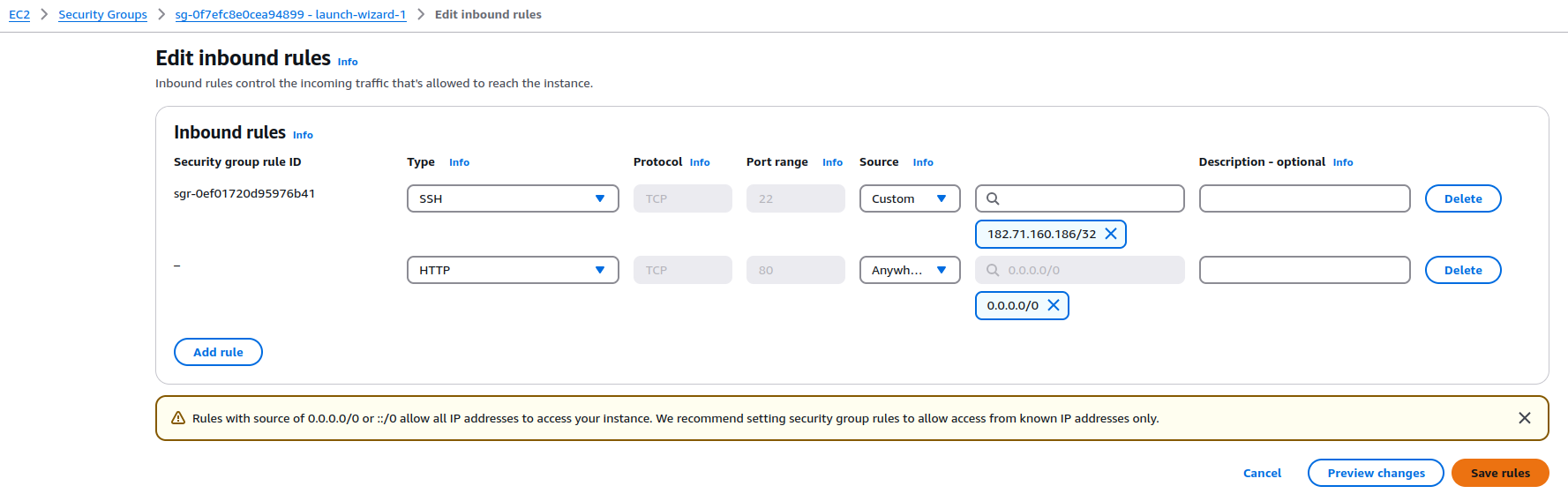
From the local machine run following commands (chmod for owner read perms only and ssh to connect to the ec2 instance) and now we are ready to use the ec2 instance

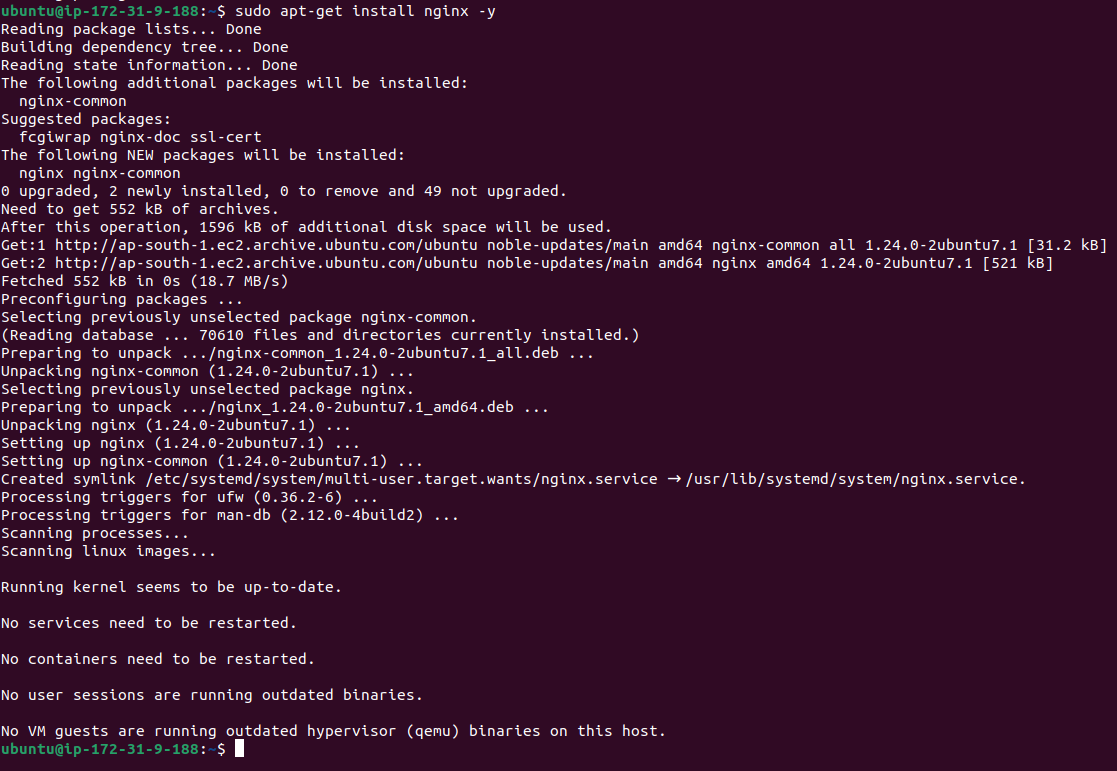


*Q4)Install nginx package in the above server and access this page from your local browser using a domain name instead of IP address of the server. (later given the information that hosting on the ip is fine as domain name would require a purchase)*

First select the EC2 instance, go to security tab and click on the link below Security Groups label

In the edit inbound rules tab click on the Add rule button, select the type as http from the dropdown and add 0.0.0.0/0 for allowing all the ips to visit the site that will be hosted

**

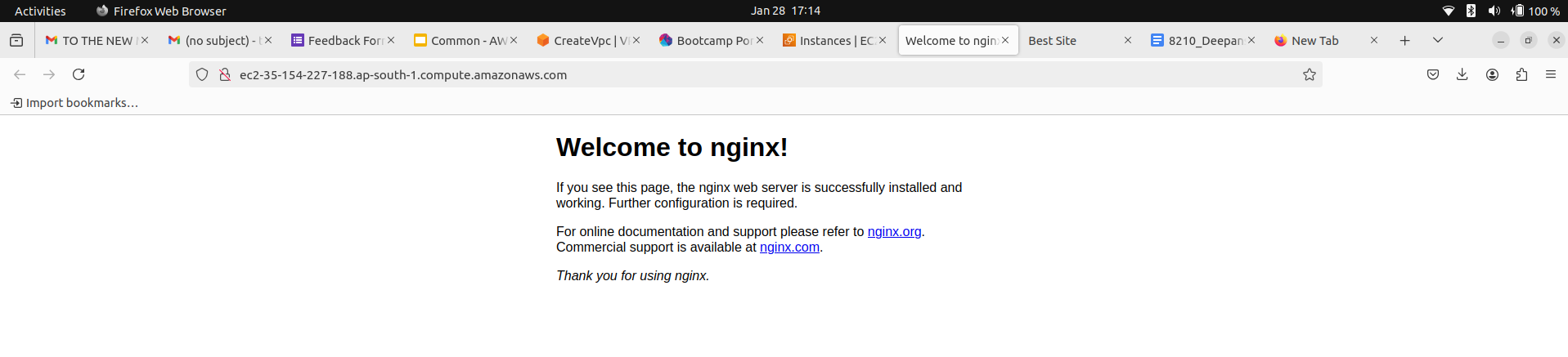
In the shell after connecting to the instance through ssh run “sudo apt-get update” to update the package manager

The run sudo apt-get install nginx -y to install the nginx package (cut from prev ss but ran systemctl status command to show the status of the package)

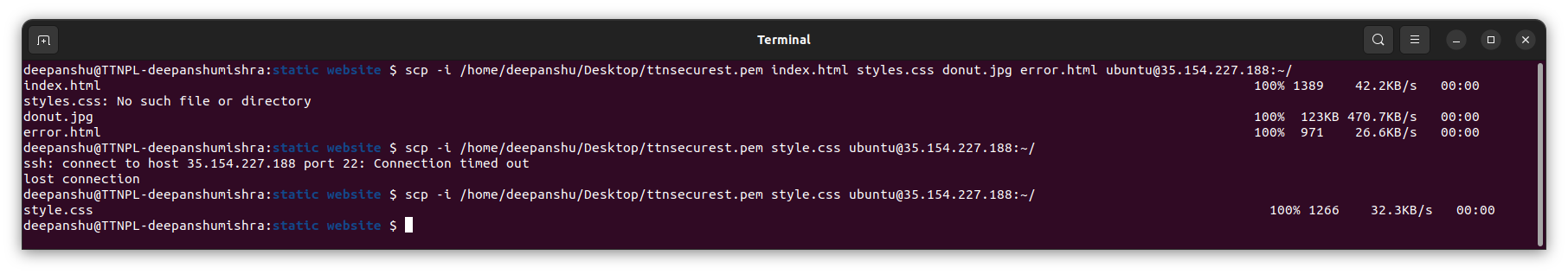
Run sudo systemctl enable nginx so that it always starts on instance startup



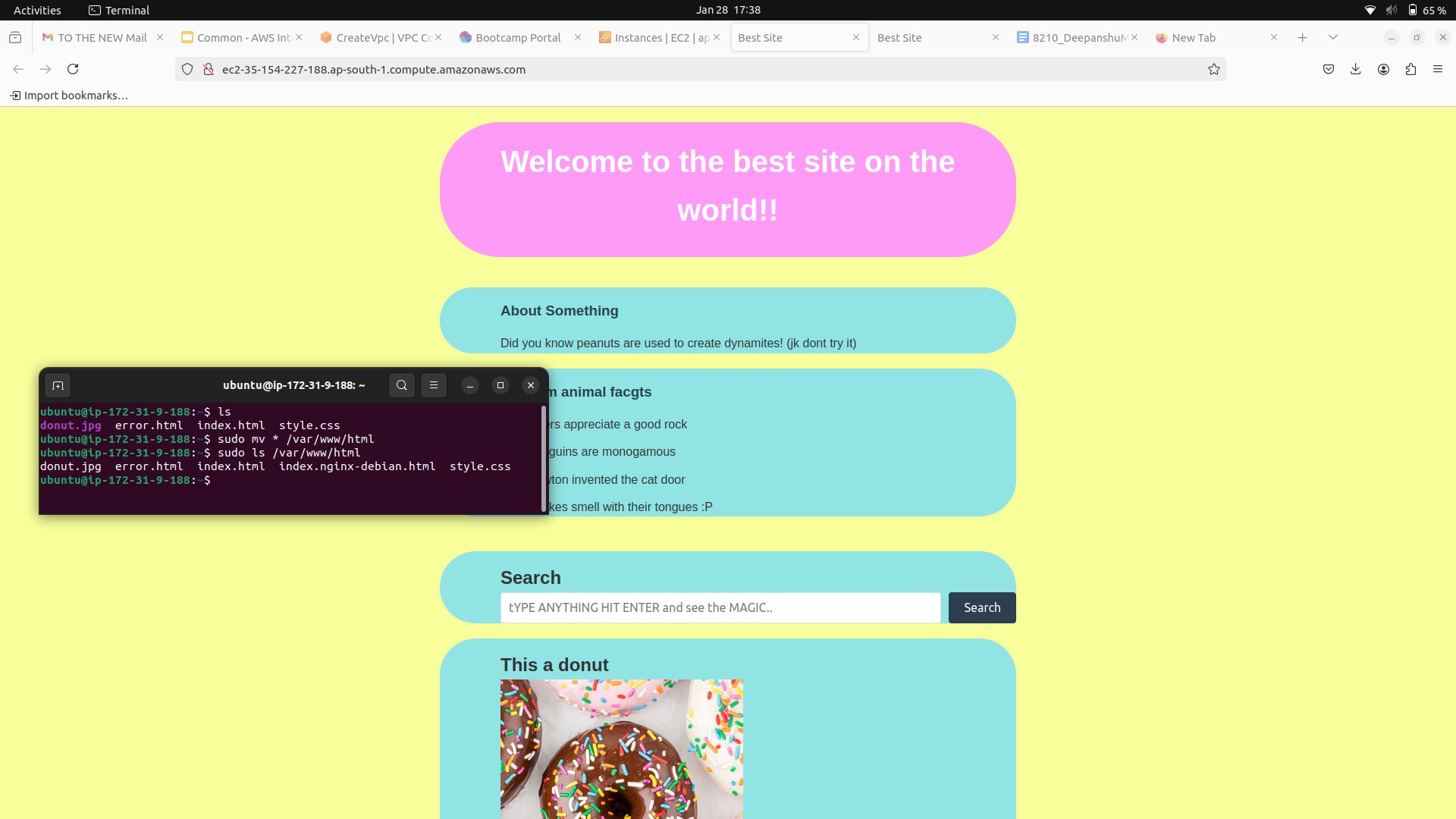
Go to the ipv4 dns in instance details to see the nginx startup screen!



Move all the static website files to instance home for deployment using the scp command from local shell

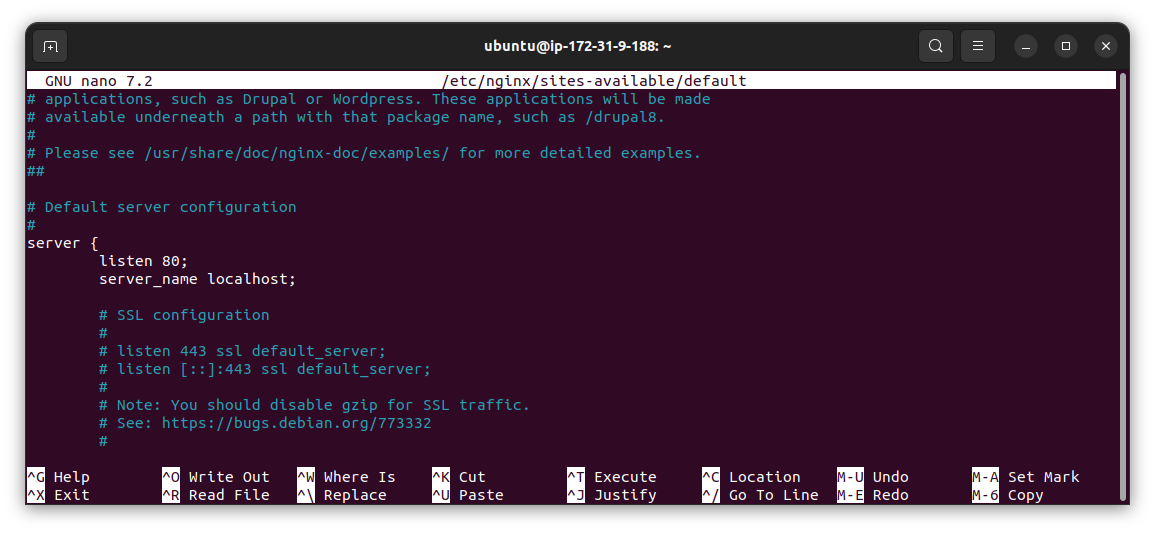


Reload the ipv4 dns to view the static website!

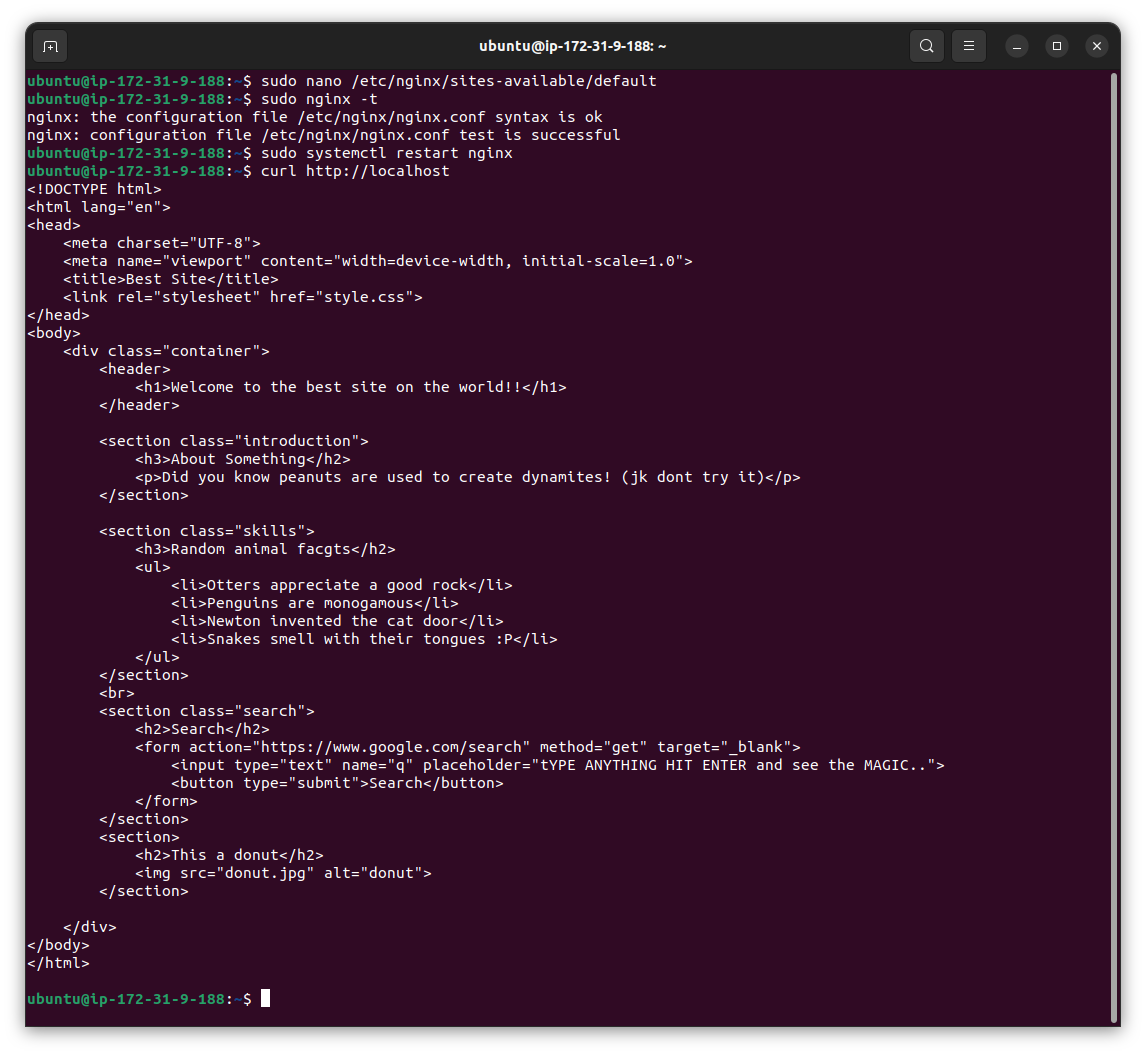


<http://ec2-35-154-227-188.ap-south-1.compute.amazonaws.com/>

Now that the site is hosted as an extra step to make sure the site is visible from EC2 graphical interface edit the default file to contain so that the site is also hosted on ec2’s localhost at port 80



Use the curl command to verify



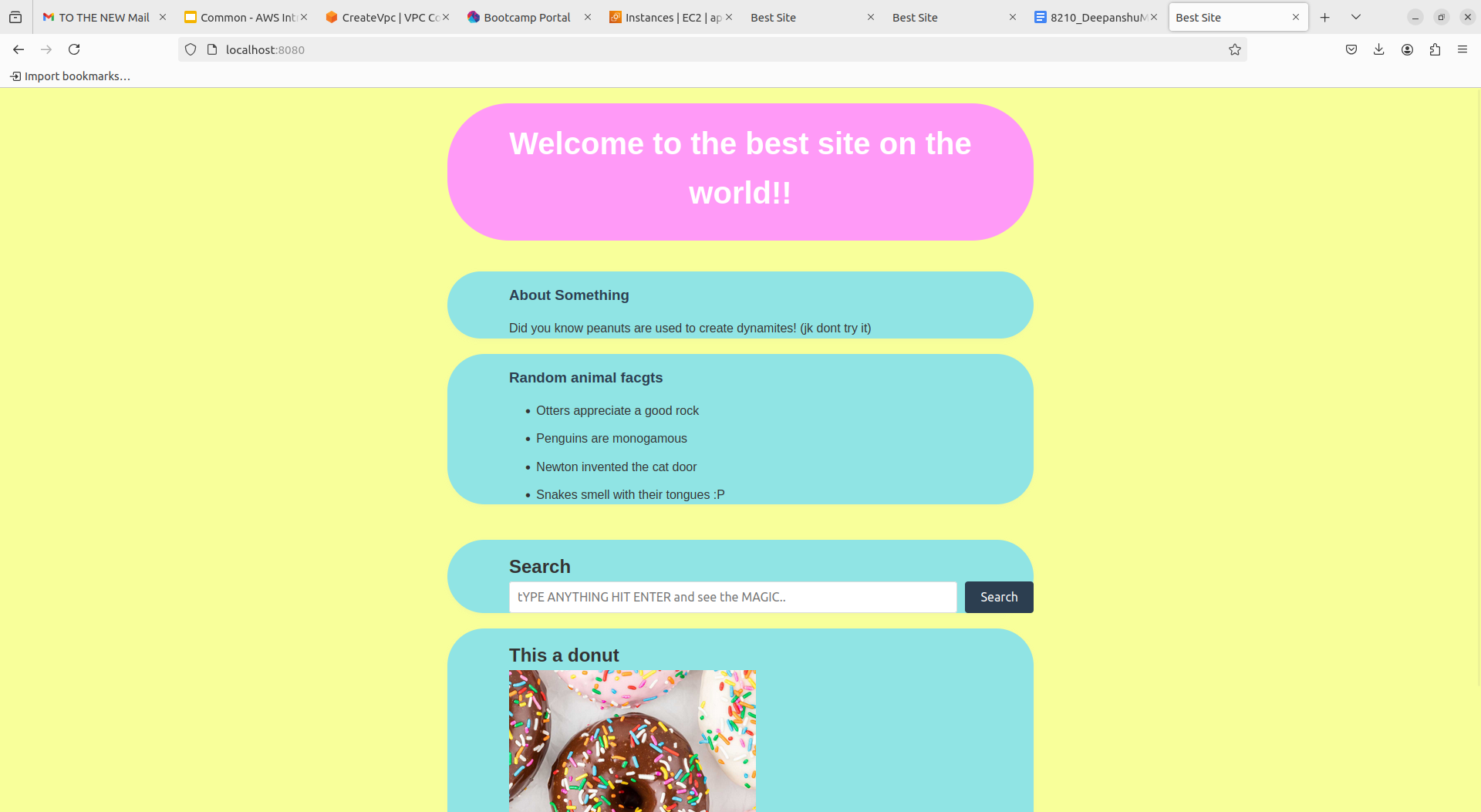
For accessing the localhost of the hosted site in the local machine’s local host lets enable port forwarding,

Run ssh -i <pem file> -L 8080:localhost:80 <public ip>

This will forward all the local machines 8080 requests to ec2’s localhost:80

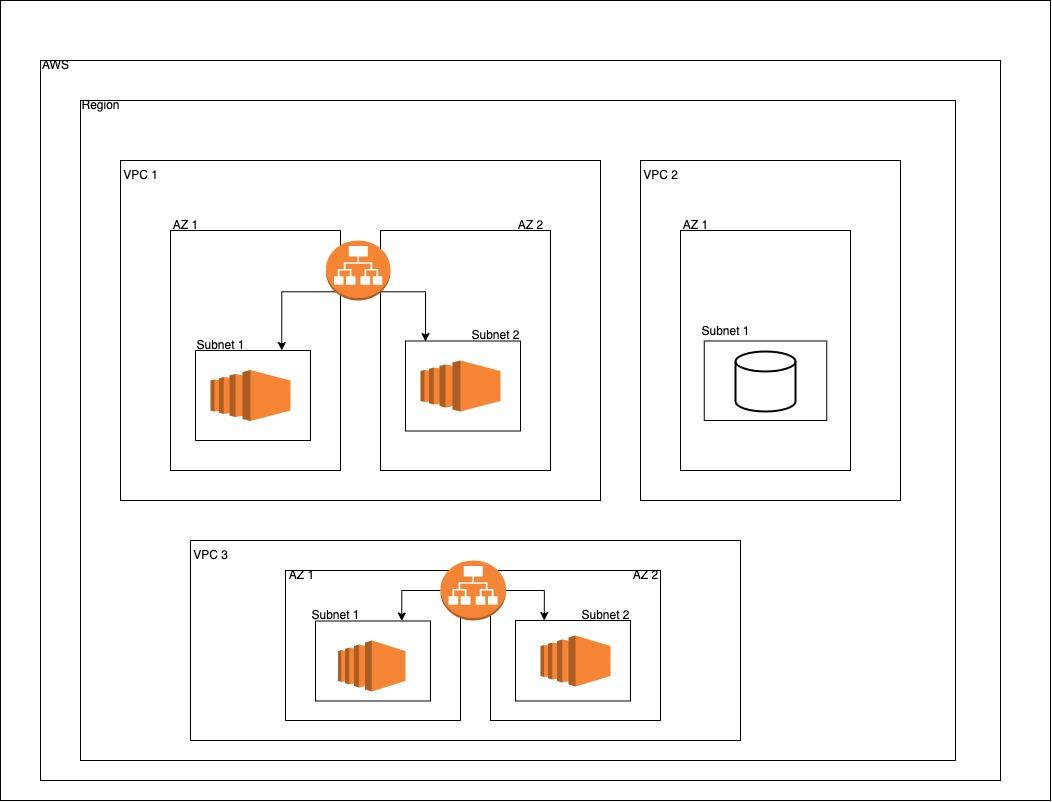


In browser goto http://localhost:8080



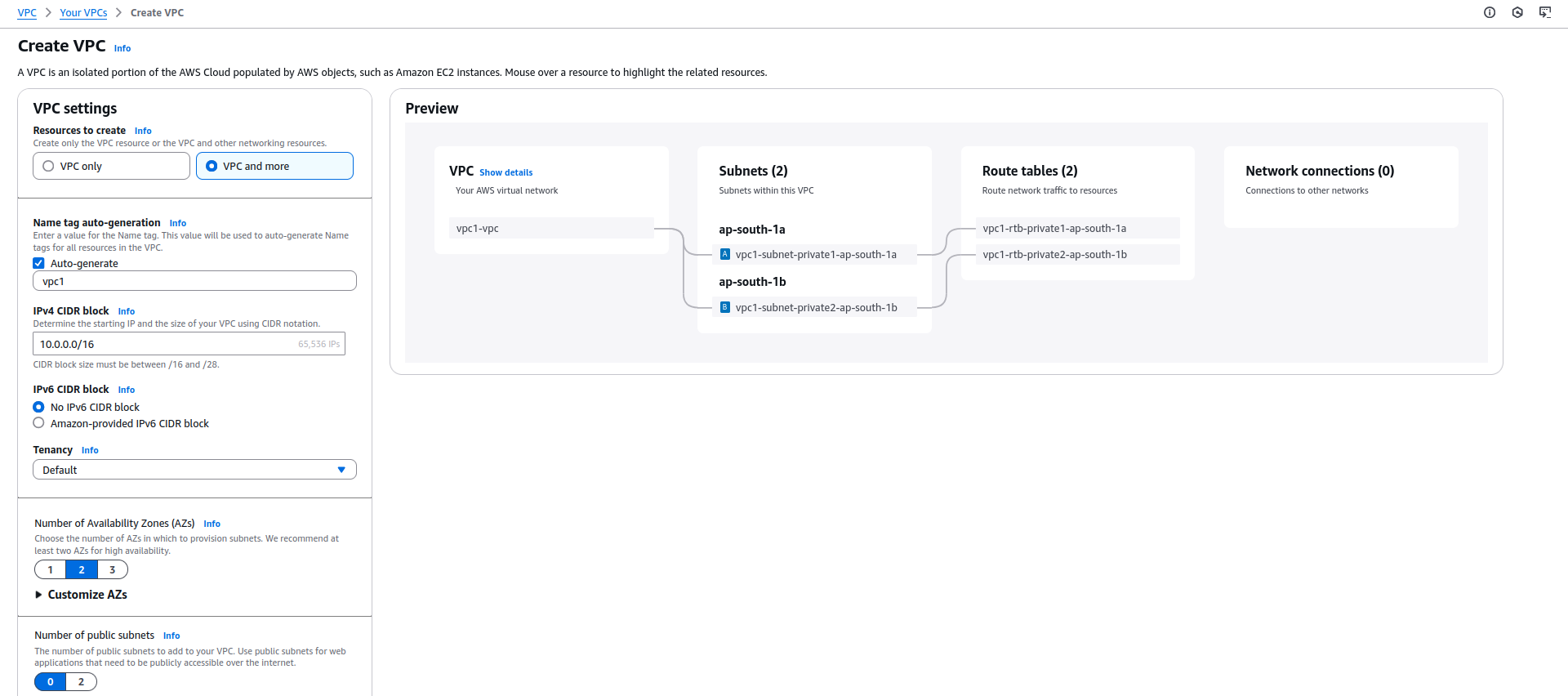
*Et voila!*

*Bonus question: Create an architecture on AWS*

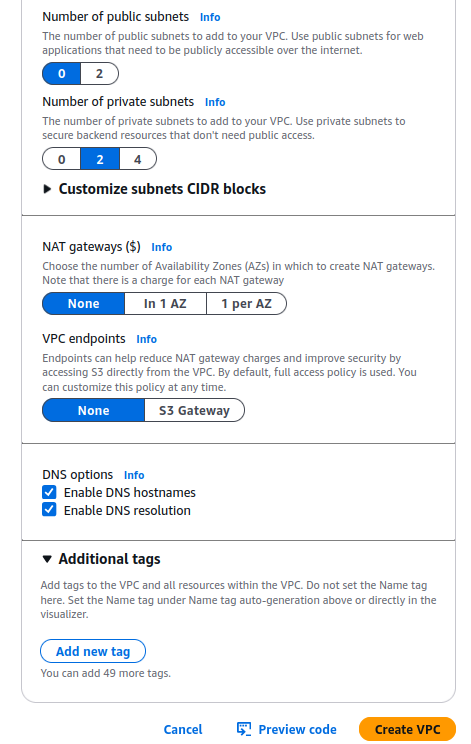
**

Login to aws console and choose a region(mumbai), go to services then vpc and add vpcs one by 1 by using the option VPC and more

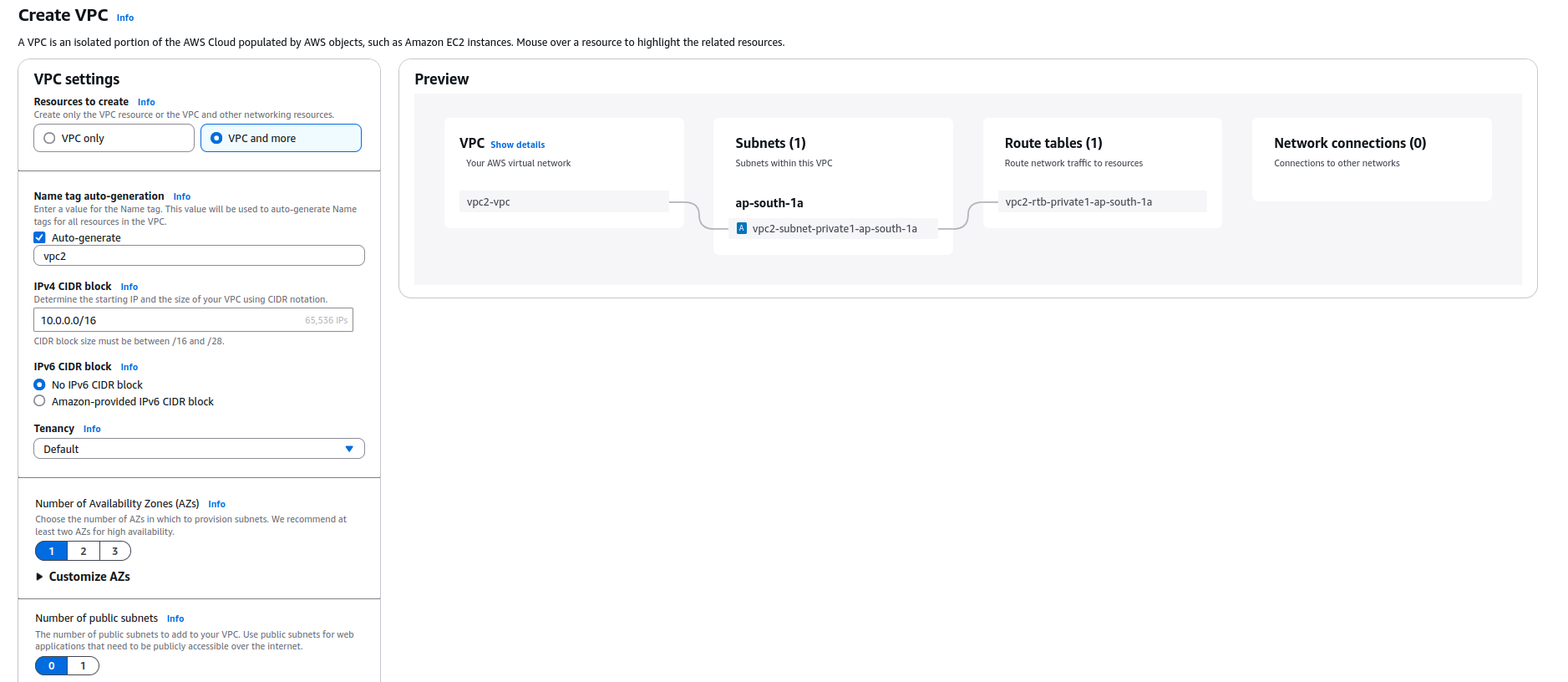
For vpc1 choose number of availability zones - 2 for high availability



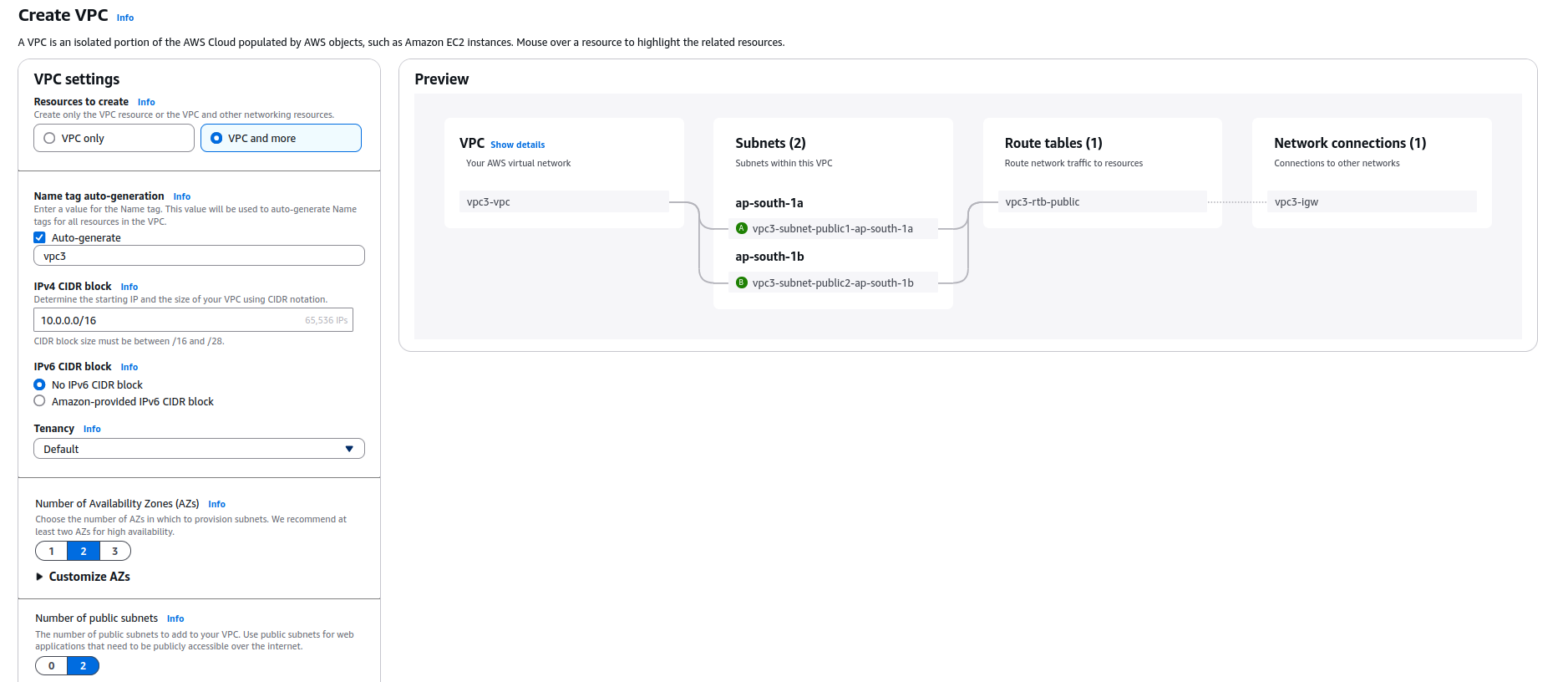
Now for vpc1 lets choose 0 public subnets and 2 private ones, choose none for gateways and endpoints and click on create vpc



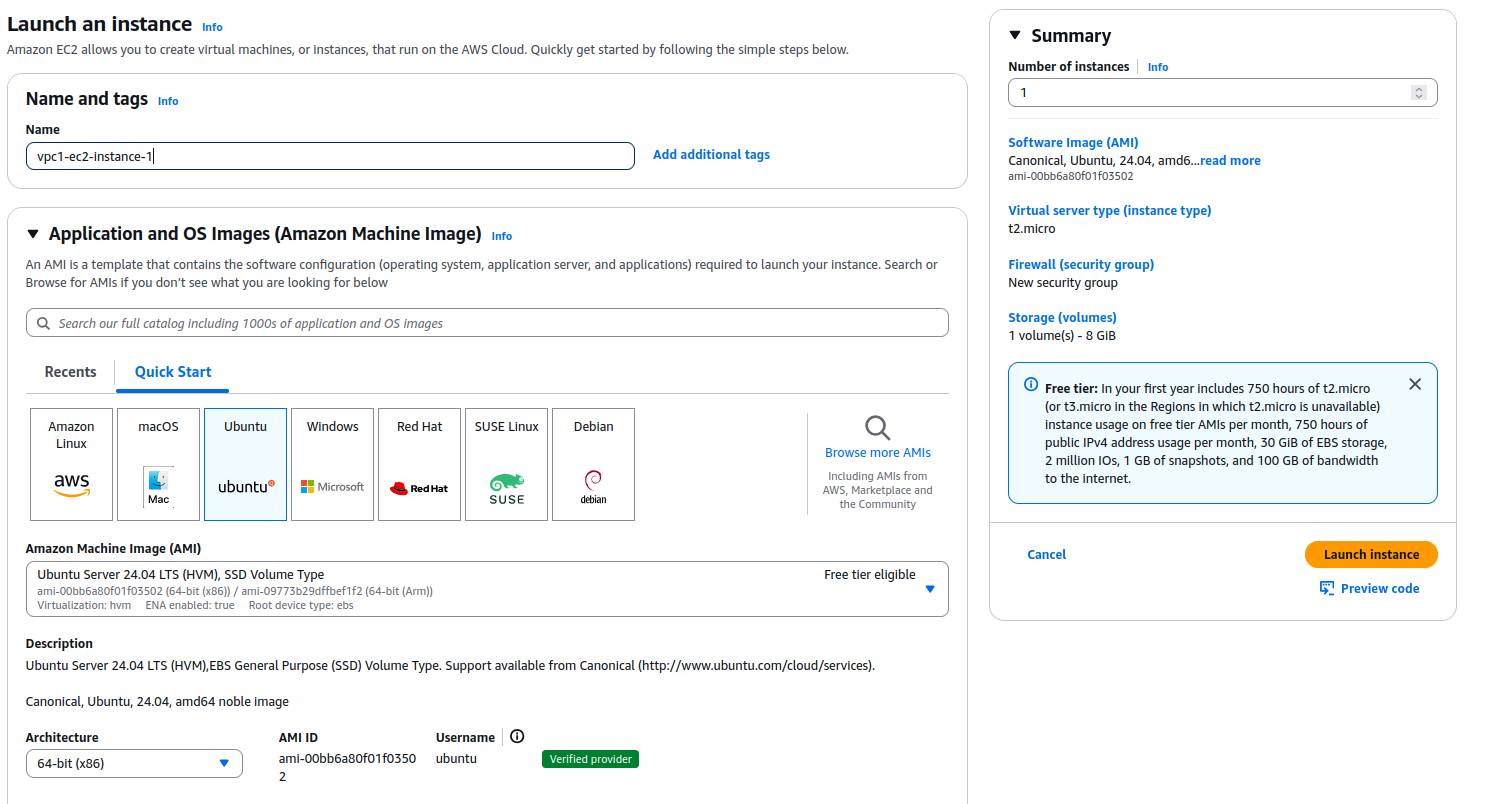
Now for vpc2 we can leave availability zones as 1 and choose same settings and create vpc



For VPC3 again lets make 2 subnets and 2 availability zones for high availability, this time choosing public subnets as 2 and private as 0.



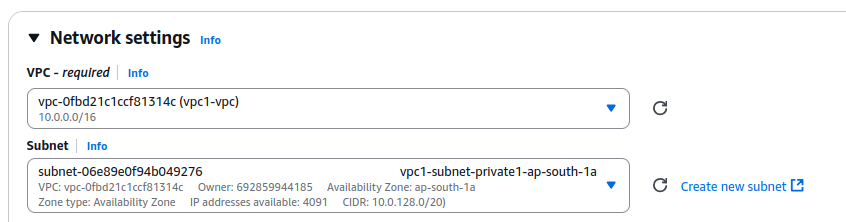
Now that we have created 3 VPCs, lets create instances for VPC1 and VPC3, name accordingly choose settings as pleased except network settings



For network settings choose a vpc from the dropdown (vpc1-vpc or vpc3-vpc) and different subnets for different zones like 1b in instance1

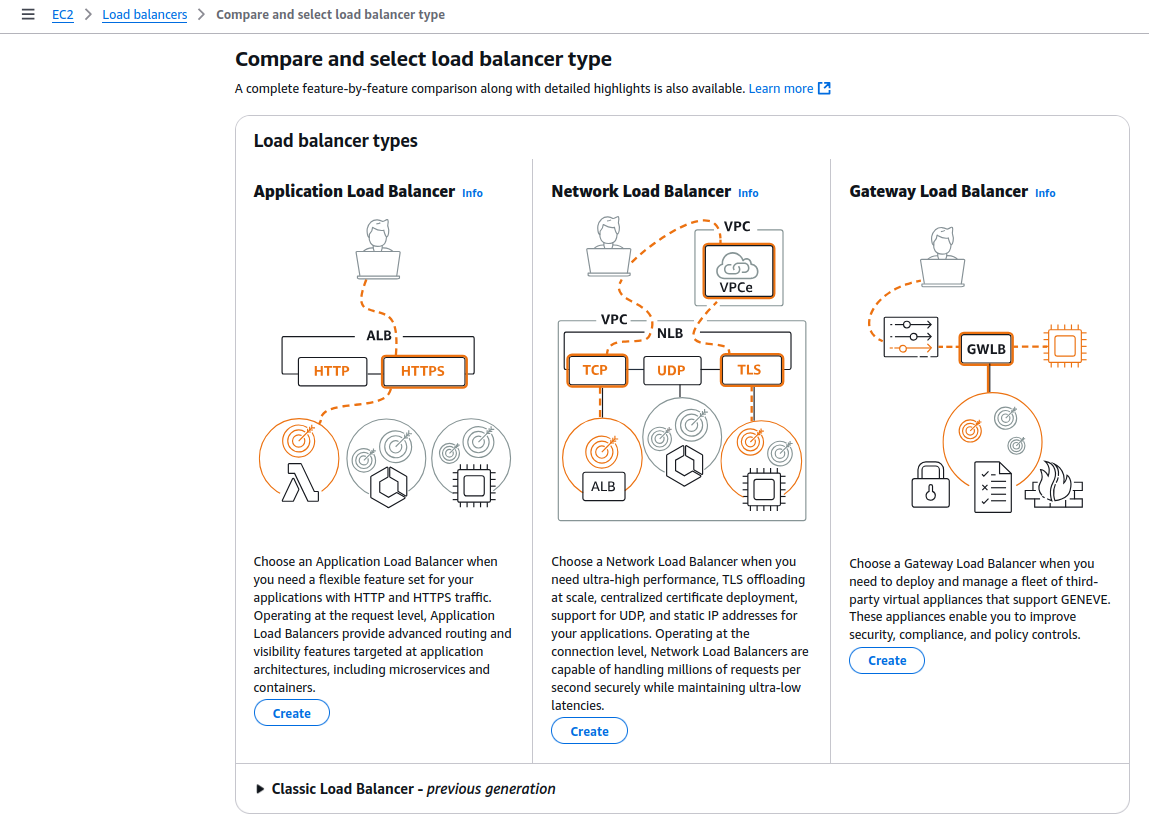


And 1a in instance2

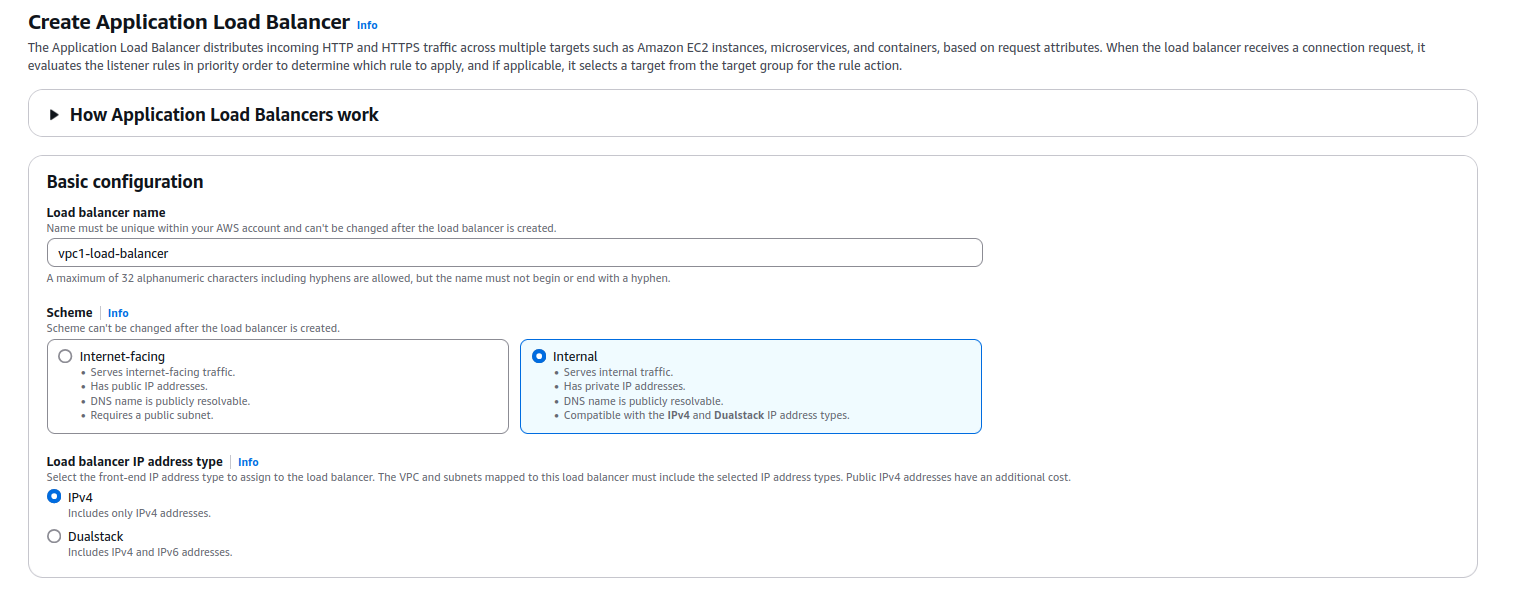


Now simply click on the launch button to create your instances

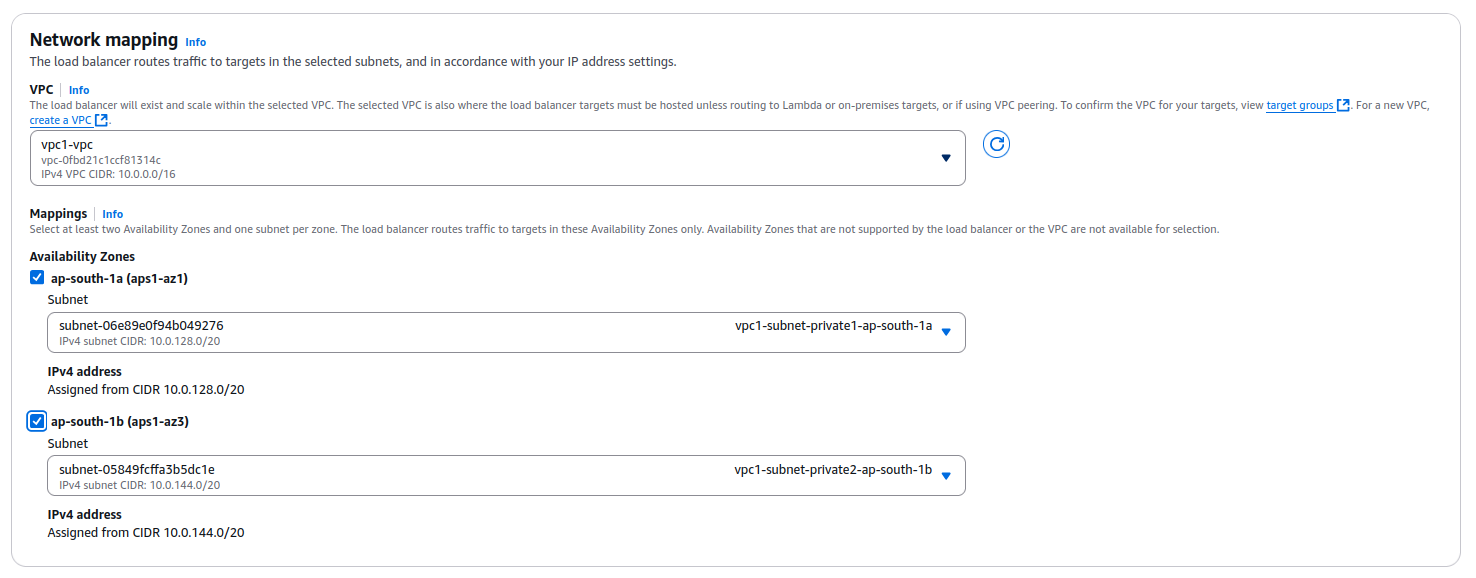
Now for VPC1 and VPC3 we need to also add load balancers for our instances, search for load balancer in aws console, goto load balancer ec2 feature and click on create load balancer, choose APPLICATION LOAD BALANCER or ALB from the choices



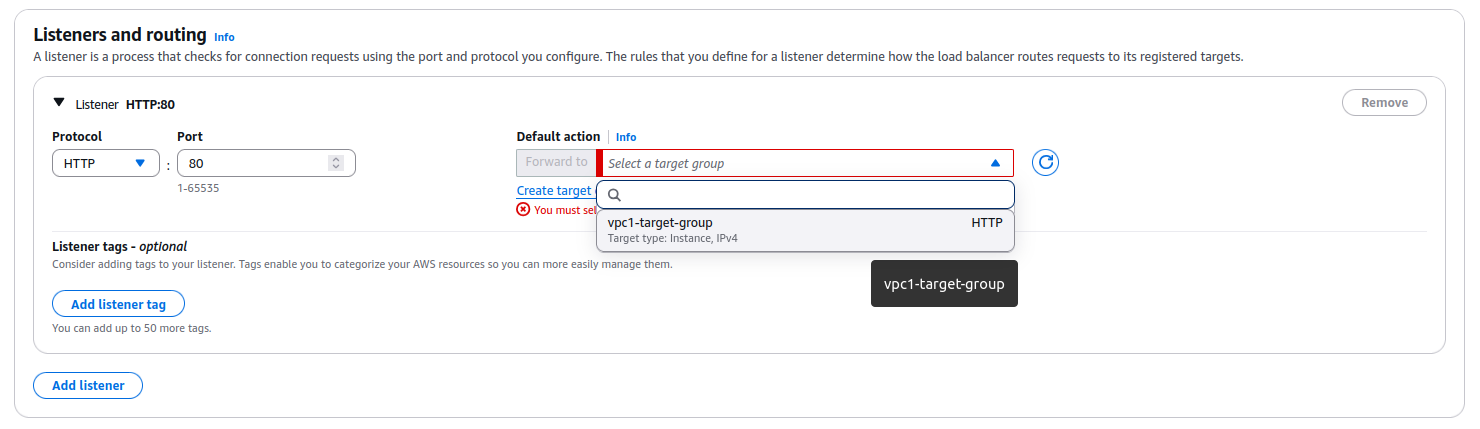
Now for VPC1 as we didnt select any public subnets choose the internal scheme



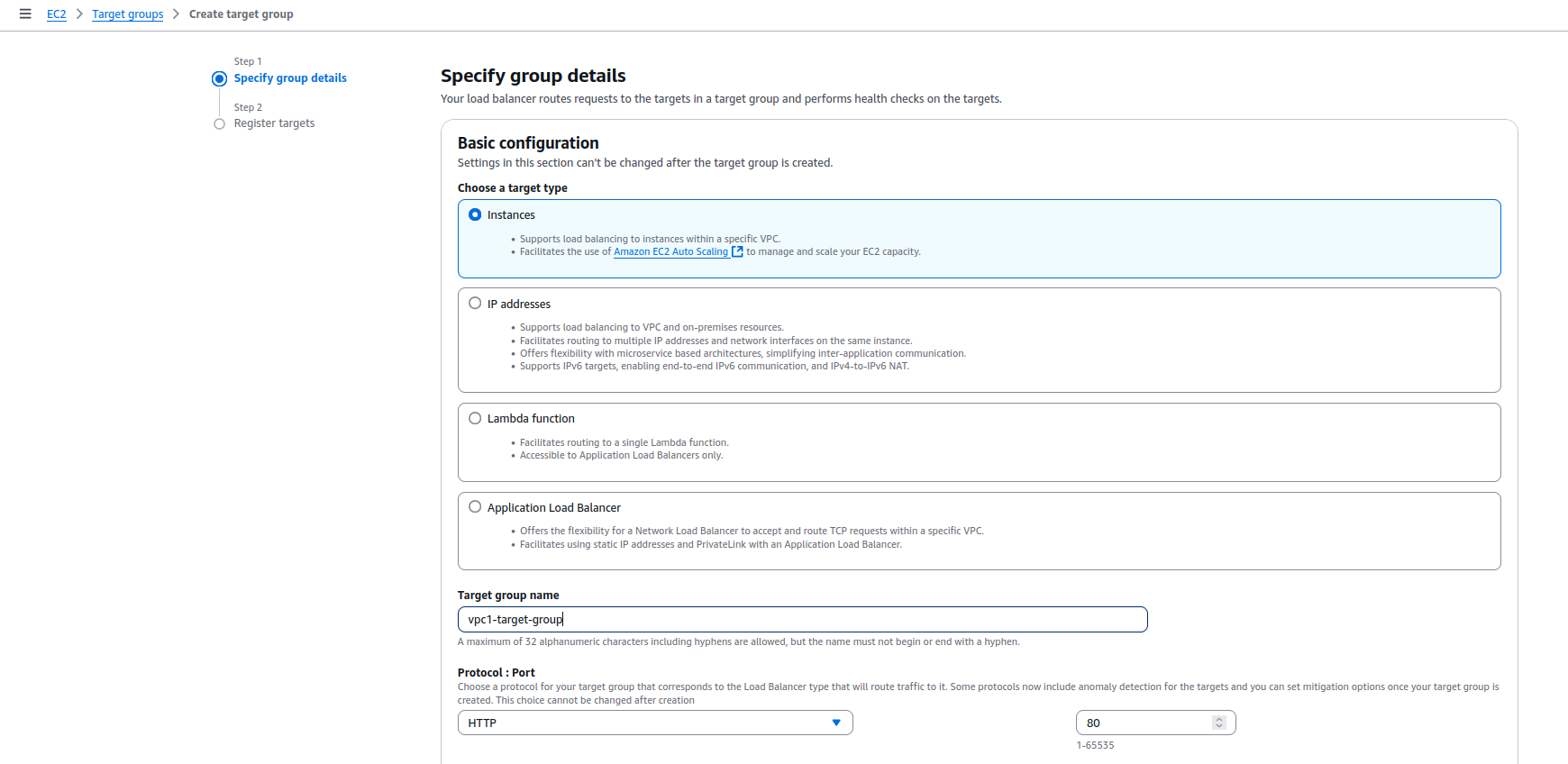
Scroll down to network mapping section and select the vpc (vpc1-vpc) from dropdown and select both the availability zones



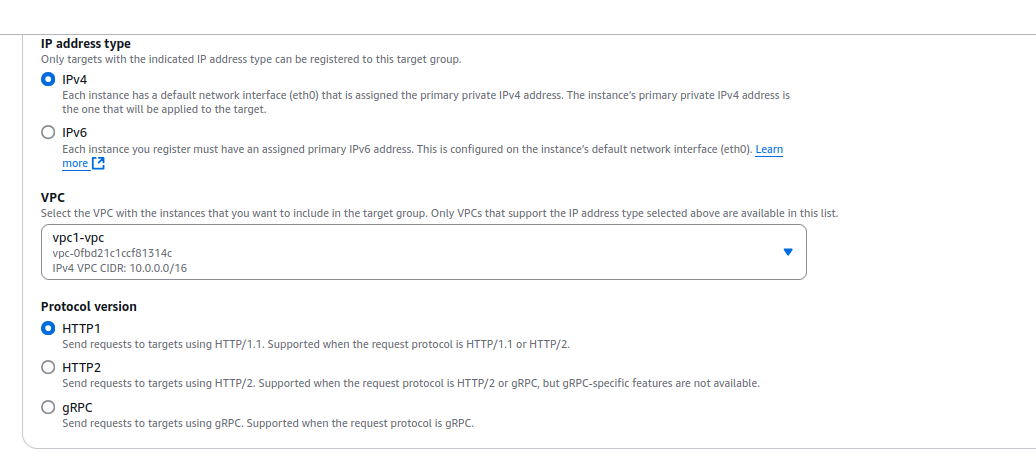
In the listeners and routing section we need to add a default forwarding action or a group, since we dont have 1 at the start we can click on the create target link or we can select a target group from the dropdown



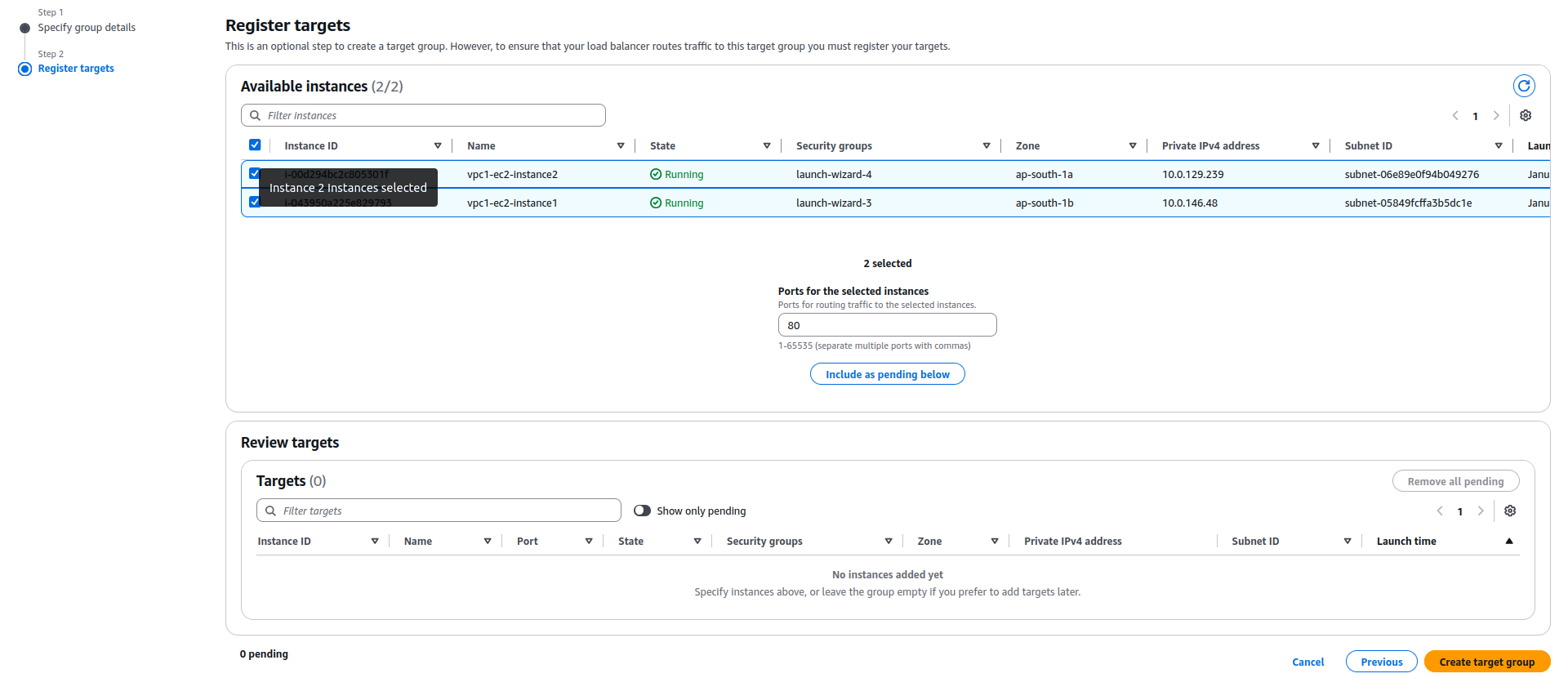
When we click on create target group link we can name the group as we please, keep the type as instances



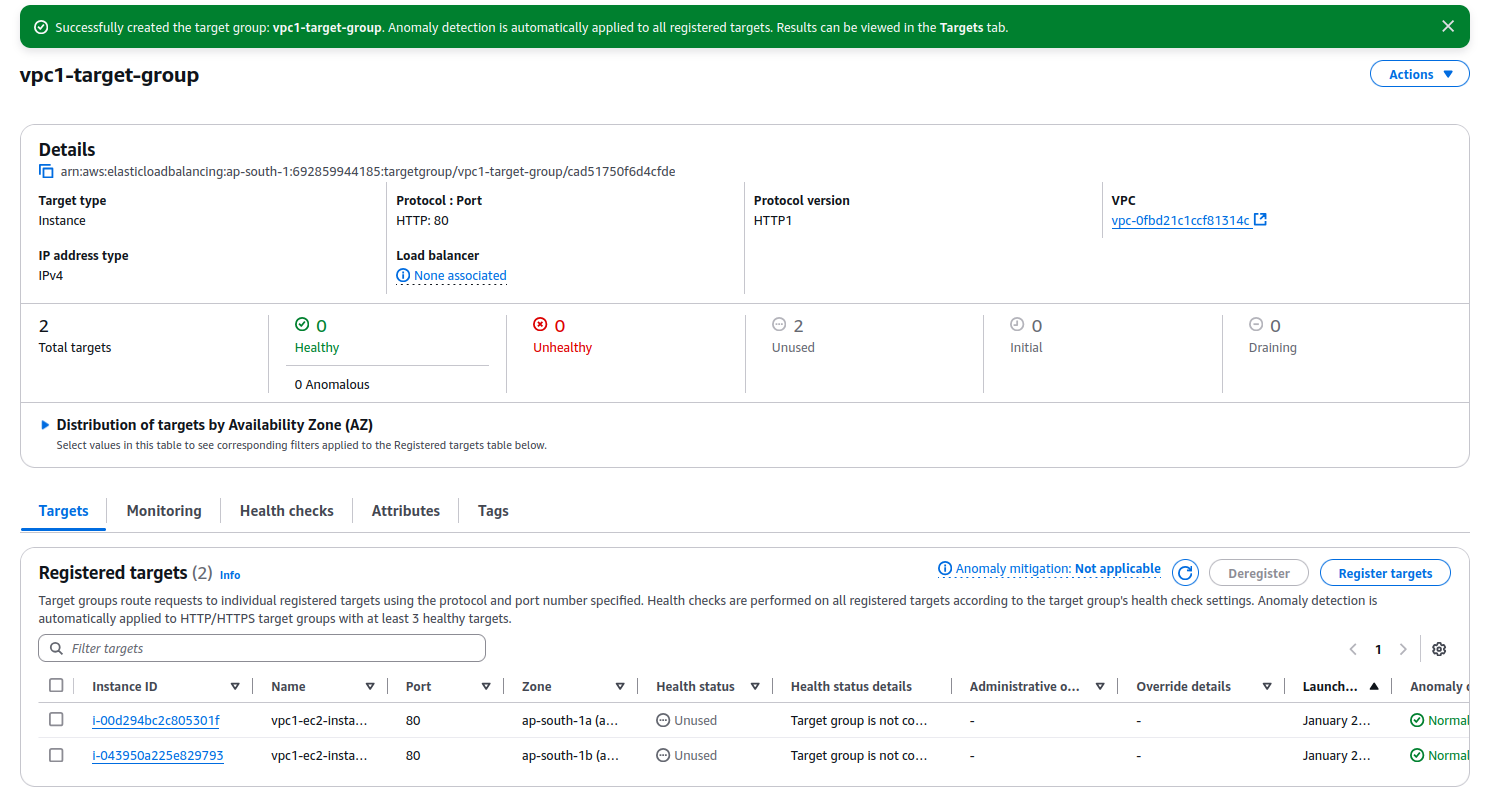
Make sure to select the correct vpc for which u want to create a target group (vpc1-vpc) and click the next button



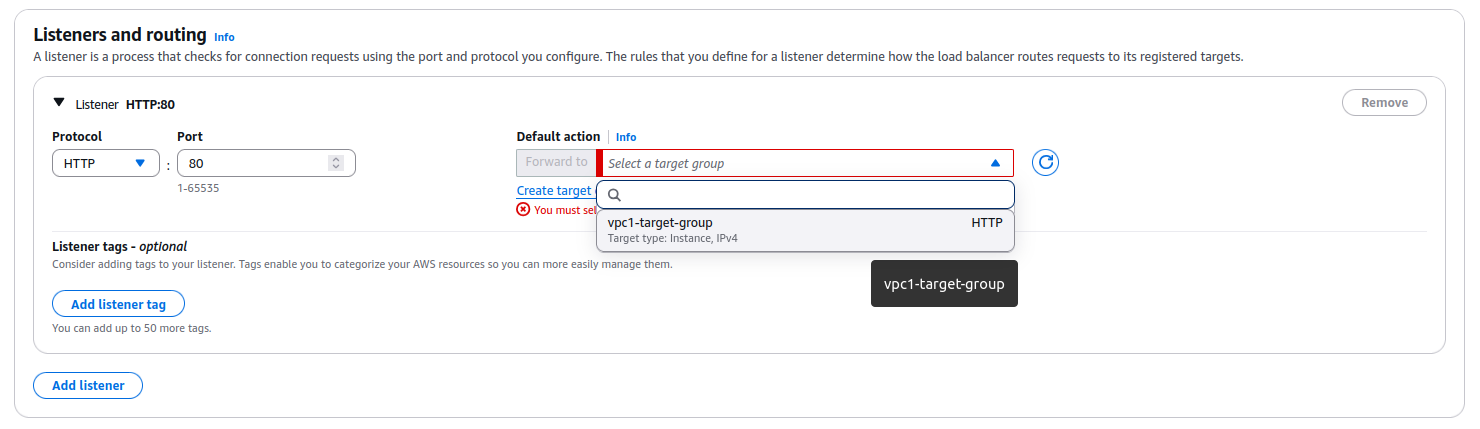
Now we need to register targets, since we only created 2 instances select both of them, click on “Include as pending below” and click on the create target group button



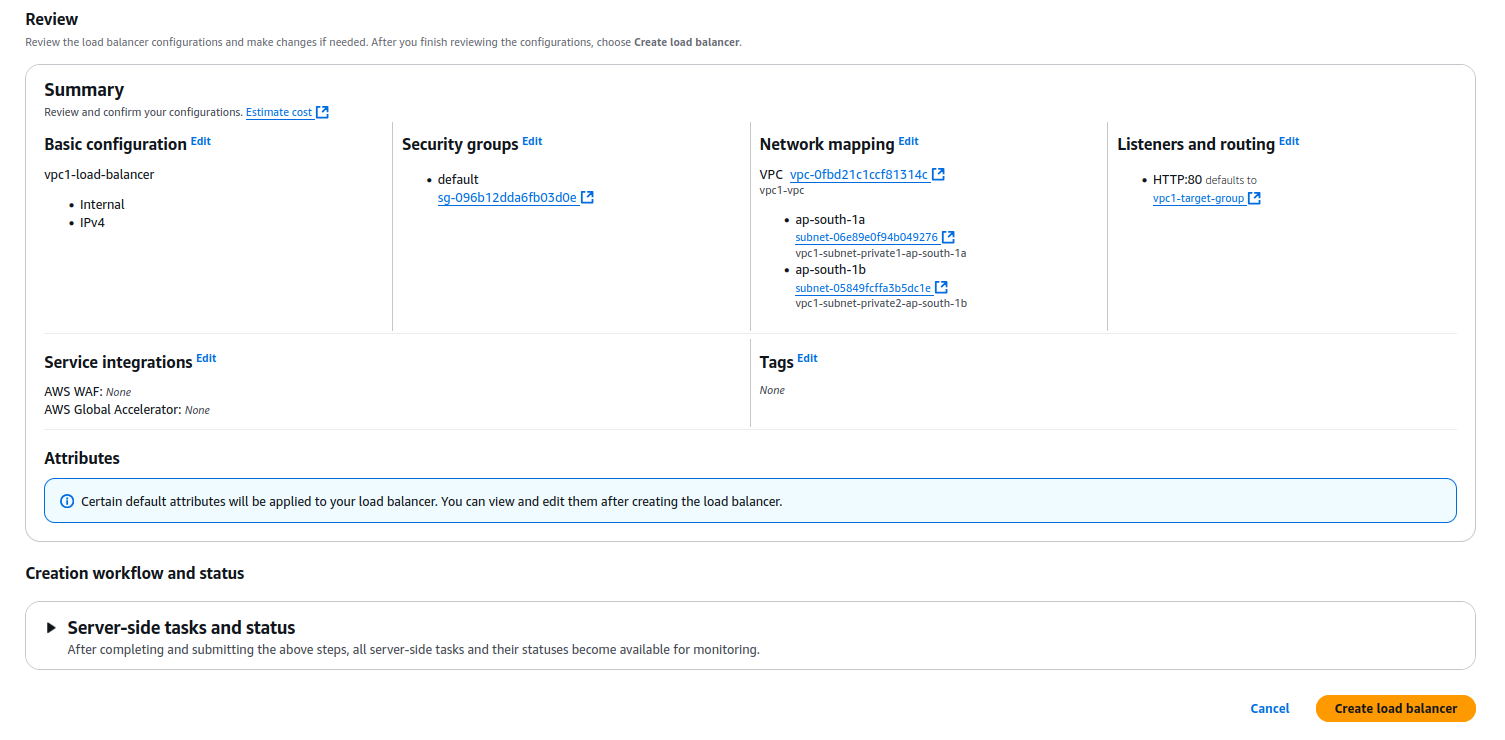
We get redirected to a confirmation page for the creation of our target group, here we can also see health of our targets and their information



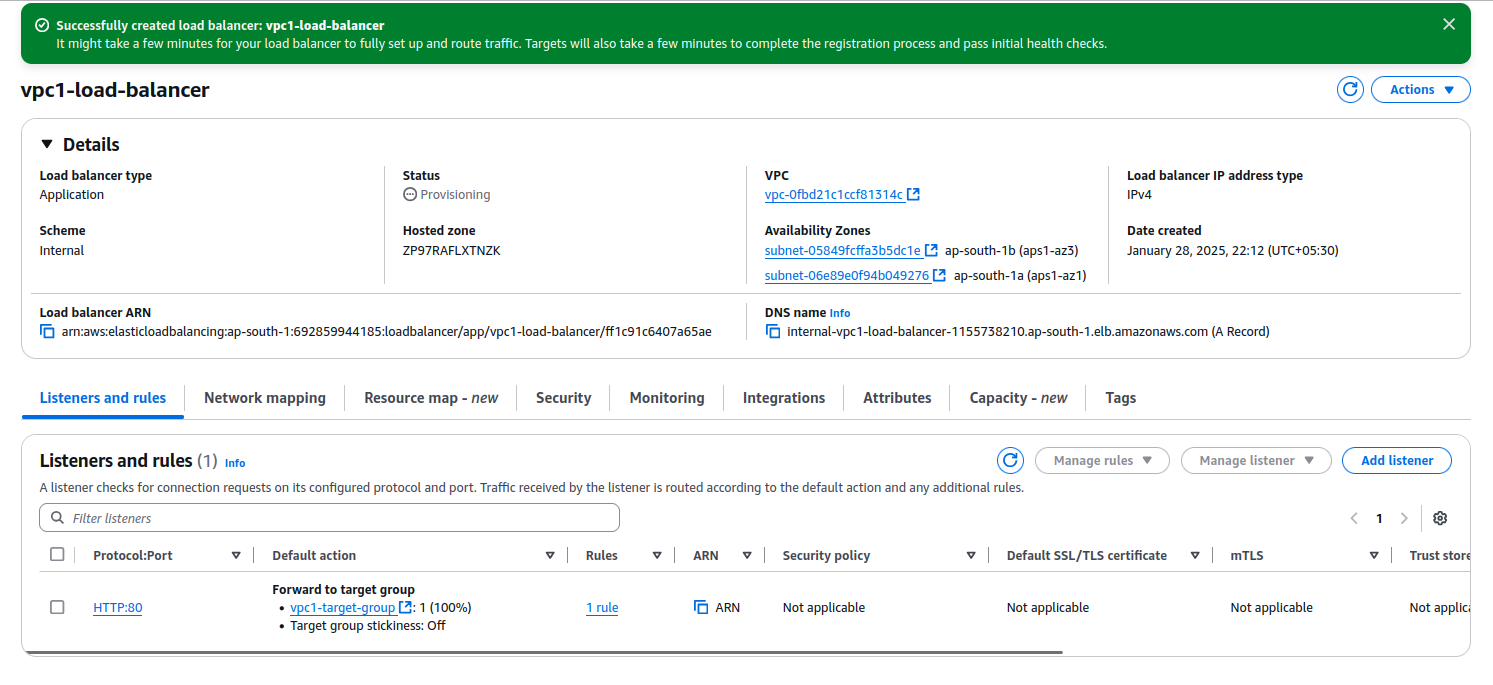
Once a group is created click on the refresh button next to the select a target group dropdown and the group should be there, select the target group.



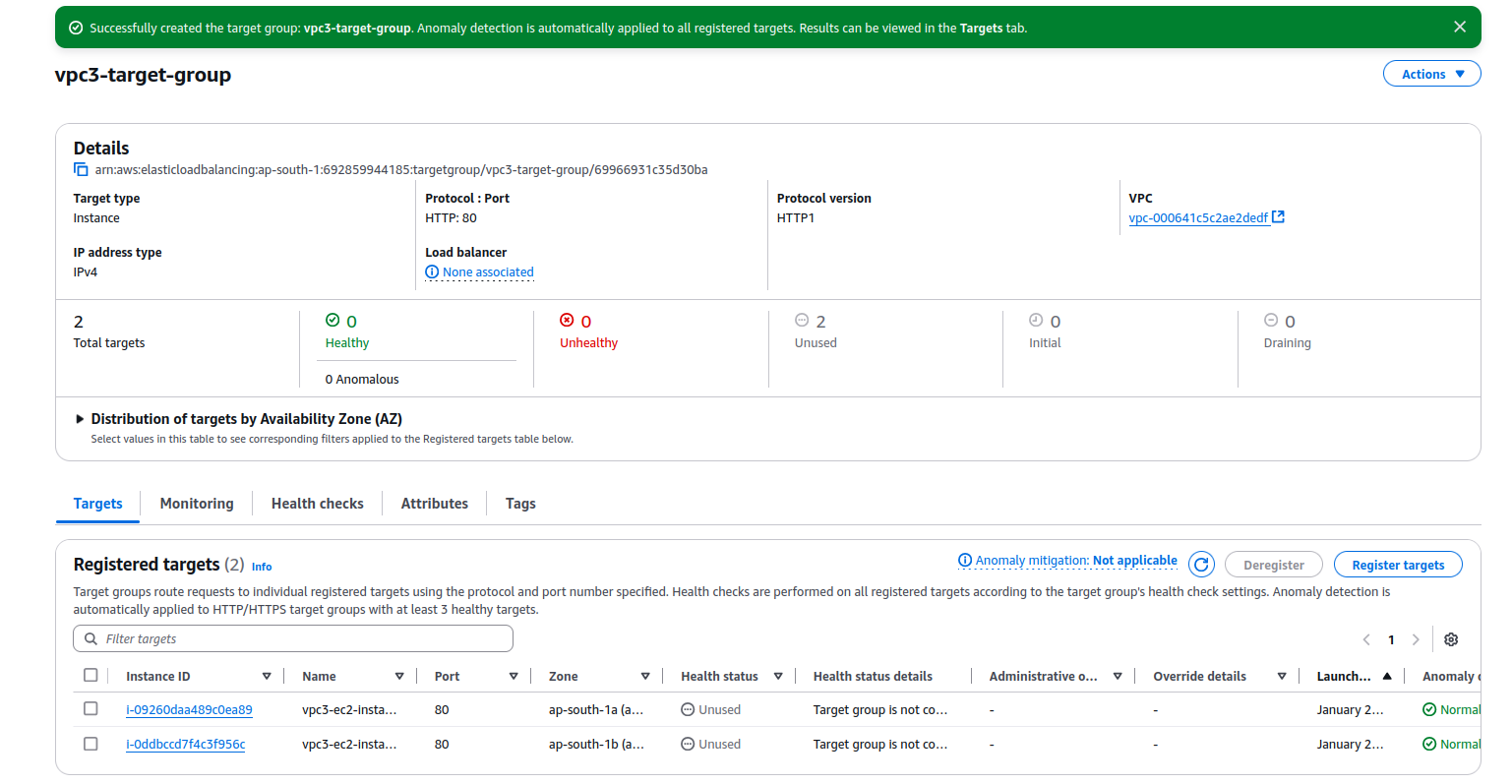
Now we can scroll down to the review section to view the summary of our load balancer and once everything’s fine click on Create load balancer button



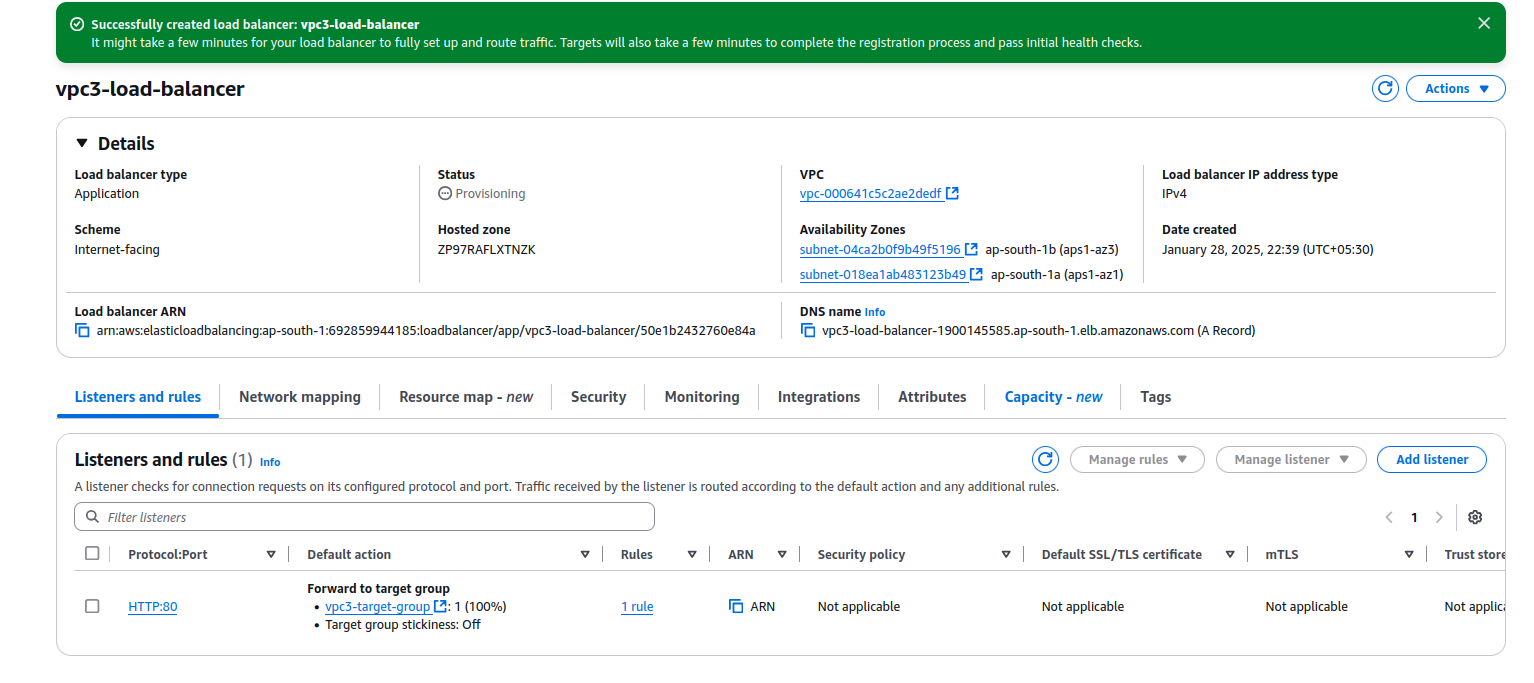
We get redirected to the details page for the load balancer with a success toast on top!



Now similarly create target group for vpc3 instances

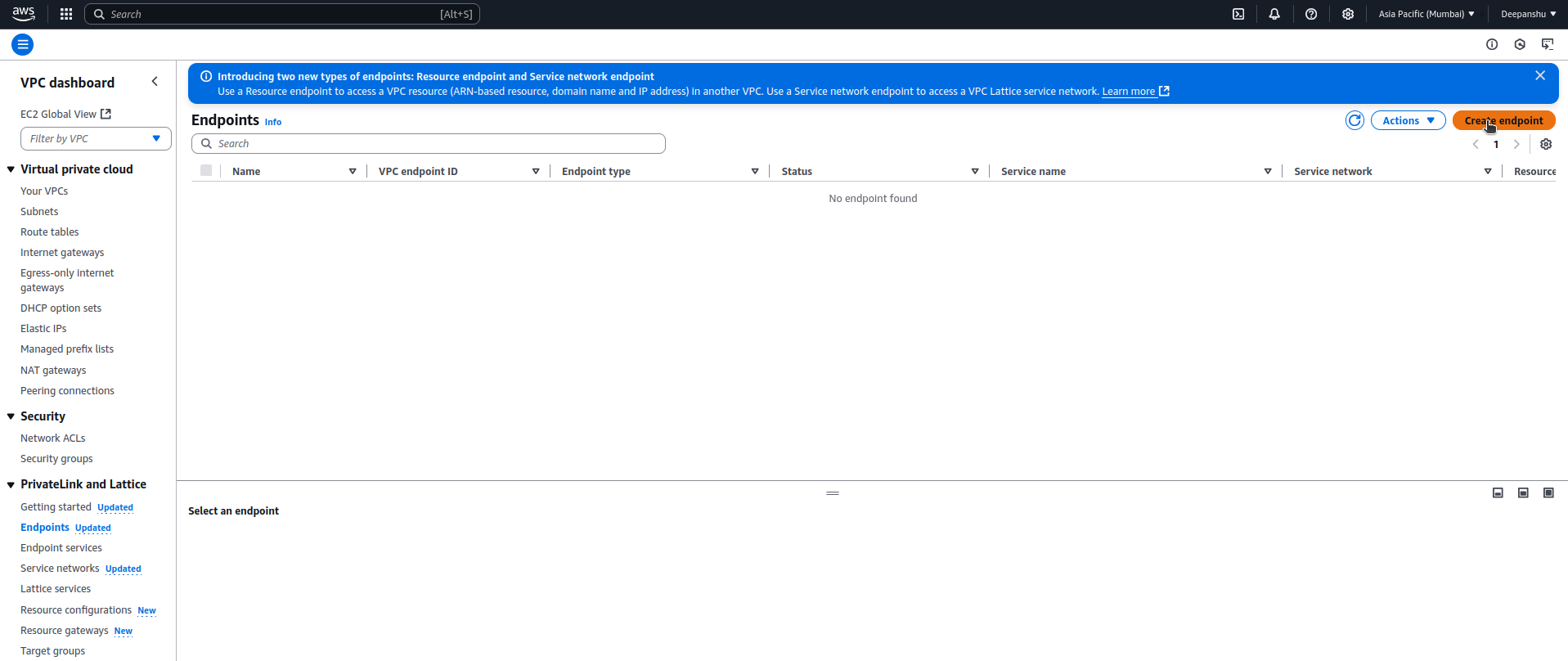


And a load balancer for vpc3

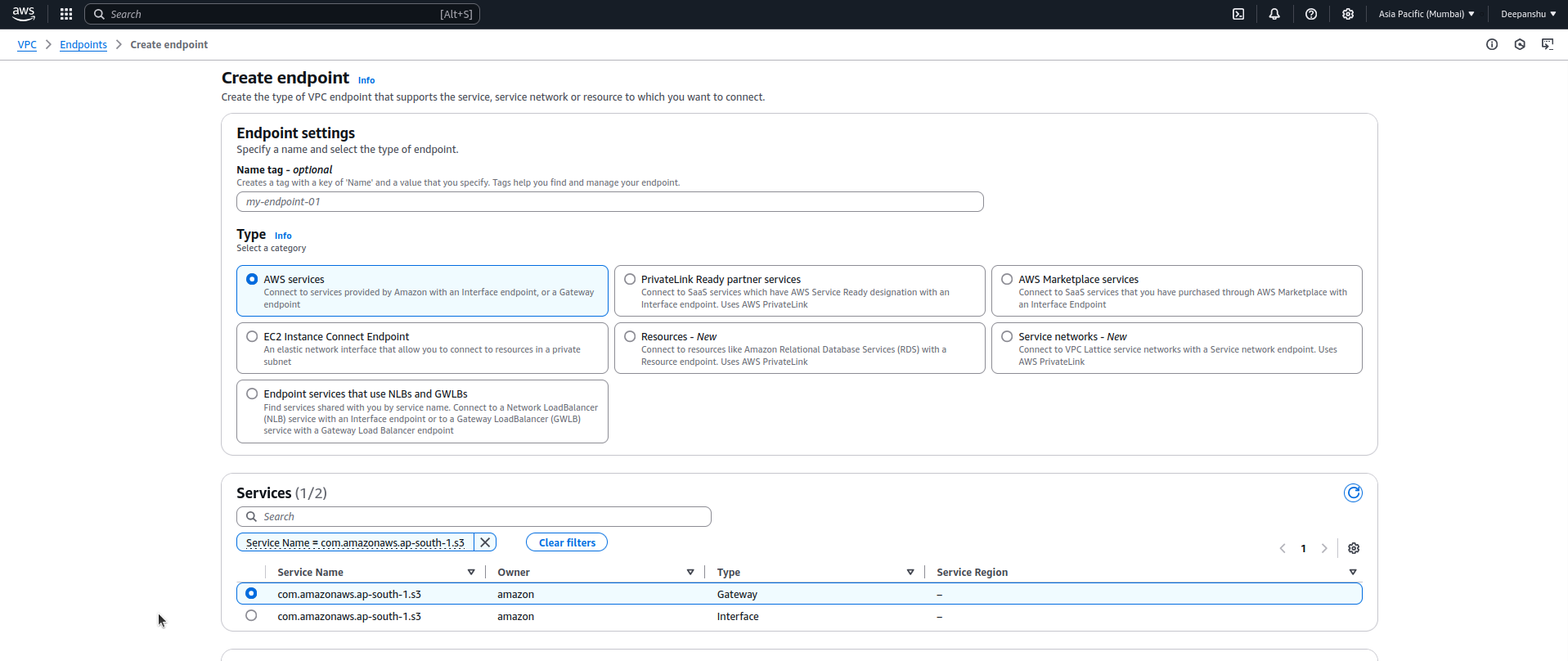


Now what resides inside vpc2 is a little unclear but lets assume we need to maybe have an s3 bucket for later use with an ec2 instance within the vpc

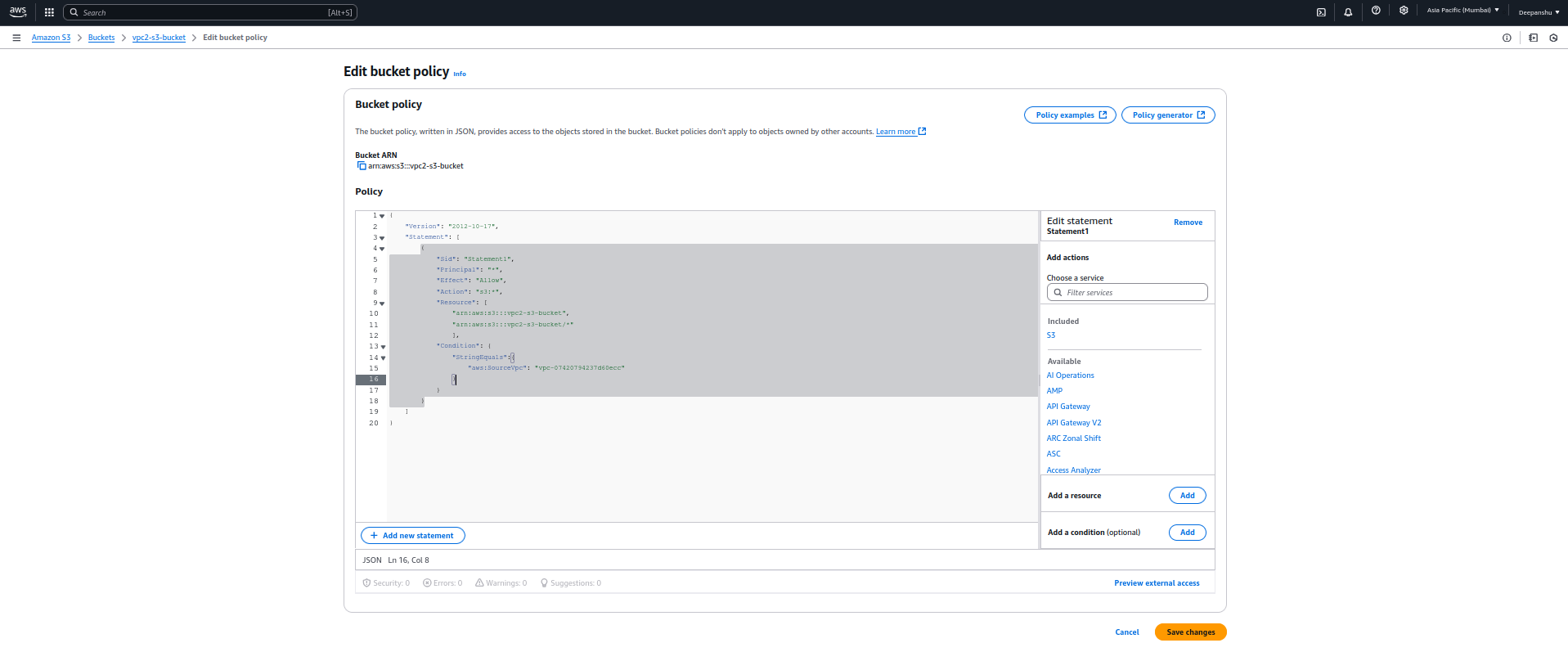
(RDS doesn’t work as in the architecture diagram vpc2 only has one subnet but RDS should have a DB subnet “group” and doesn’t allow me to initialize an RDS server) so the only remaining choices are dynamoDB or s3 bucket (for direct aws services) or we could host a postgre or mysql server within the vpc but that also doesnt align with the architecture given.  
  
In your VPC dashboard, look for “endpoints” in the left menu, go to this page and click on create endpoint



Select the service and region by searching for it in services section and just create the endpoint.

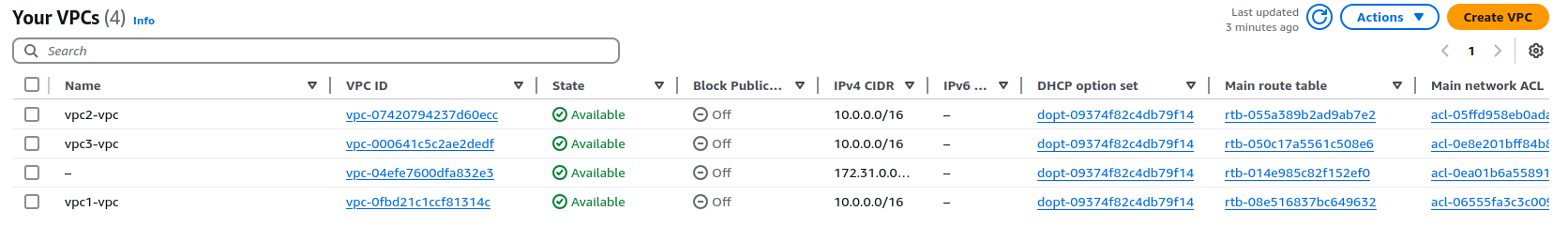


After creating the endpoint, note down the vpc id for vpc2 and create an s3 bucket, go to permissions, then go to edit bucket policy and write followin to allow only the vcp2 managed things can do operations on the objects

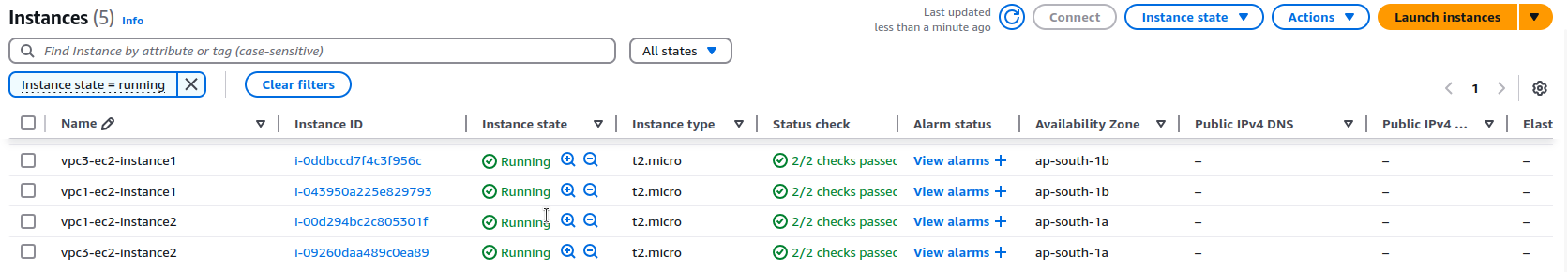


After the bonus question

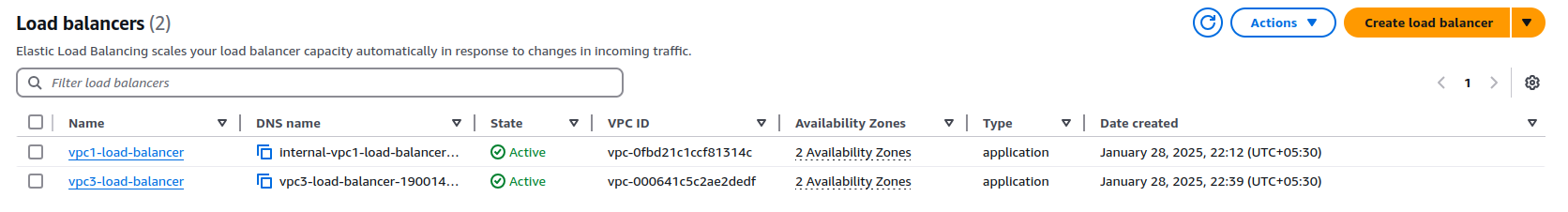
Your VPCs screen:



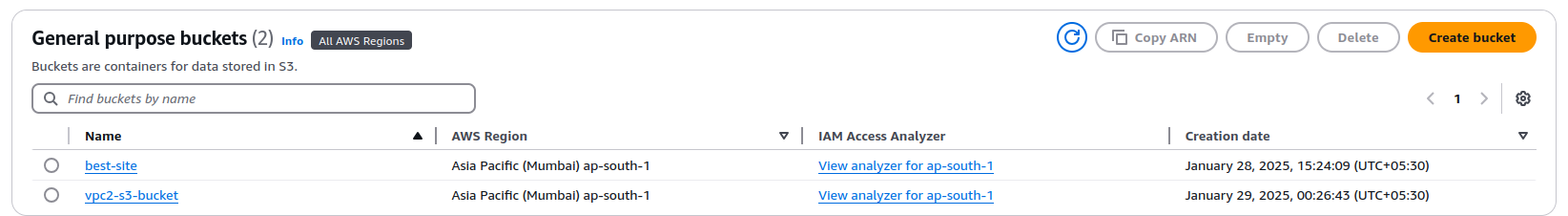
Instances screen



Load balancers



S3 buckets



Subnets

