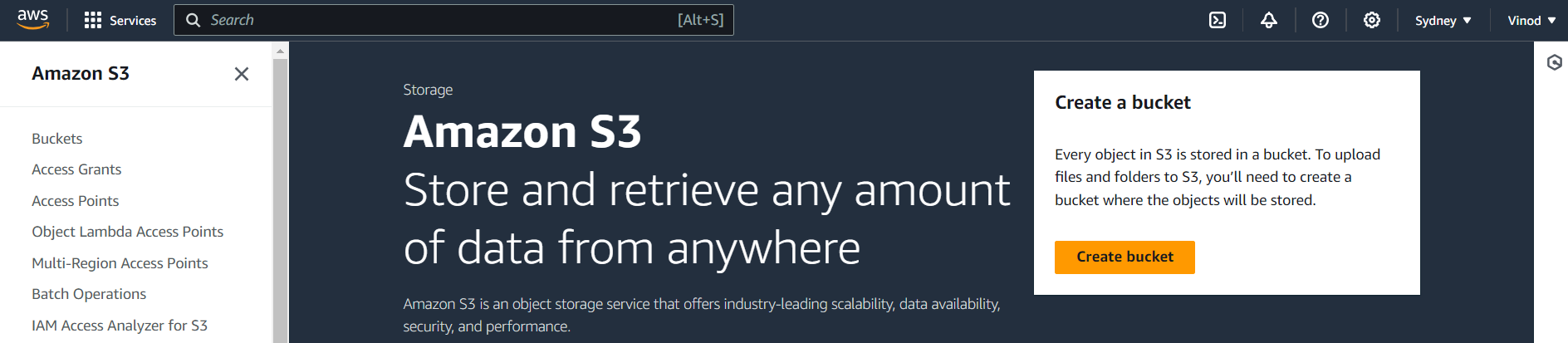
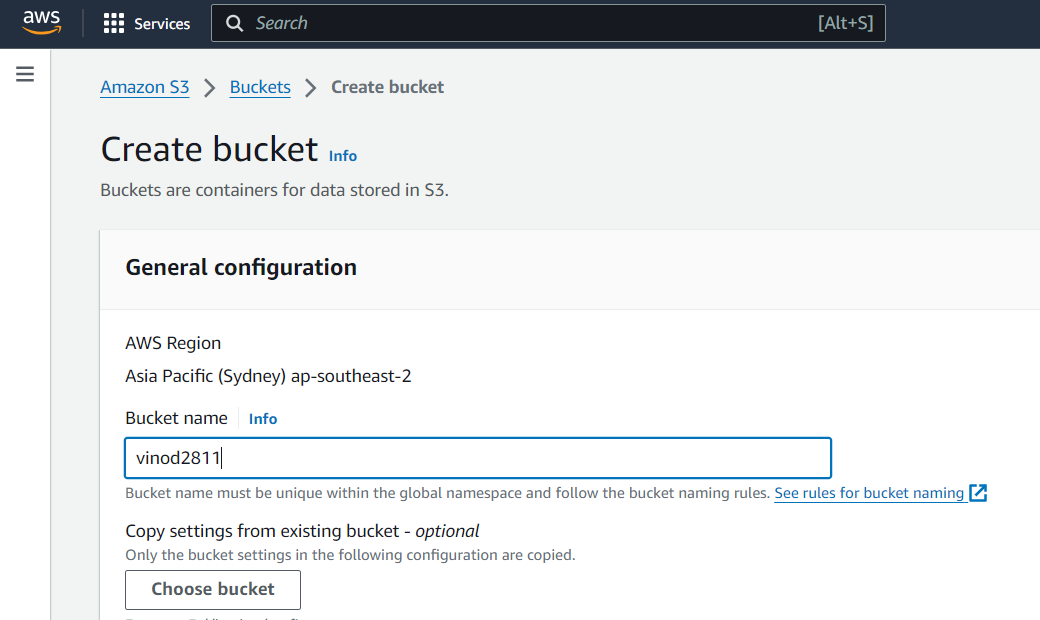
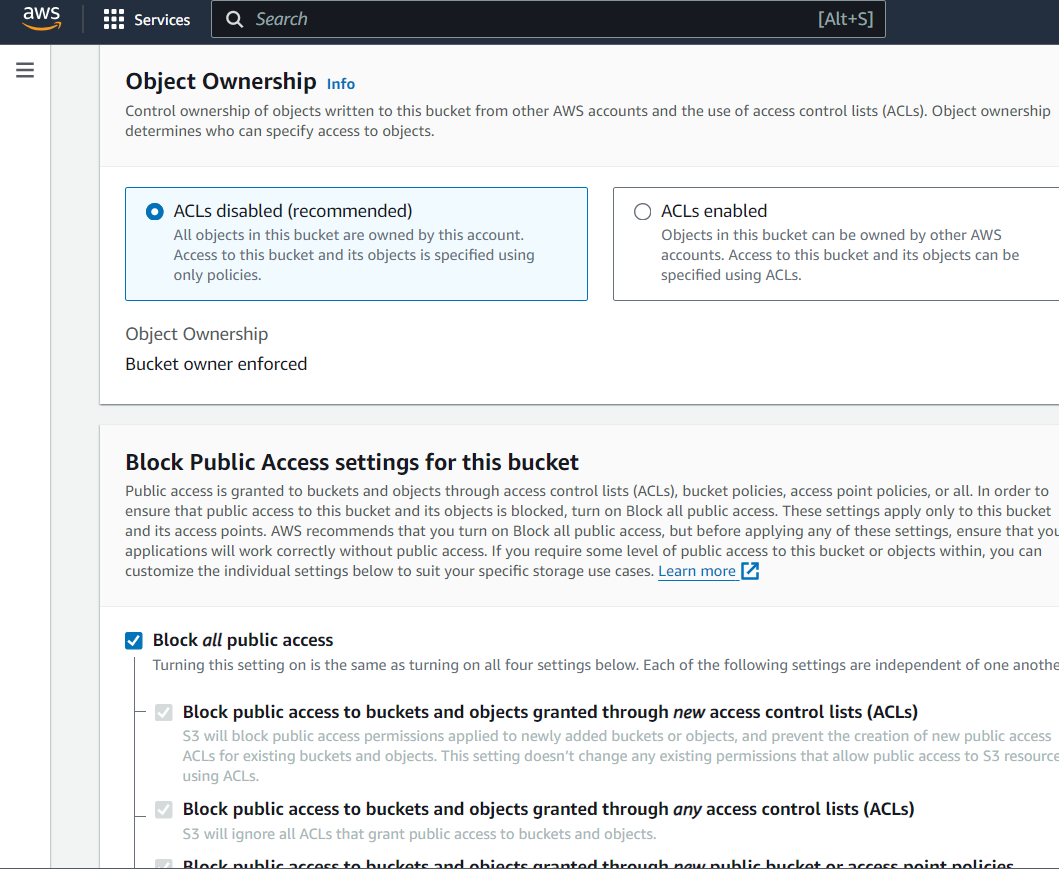
**Day 21\_task**

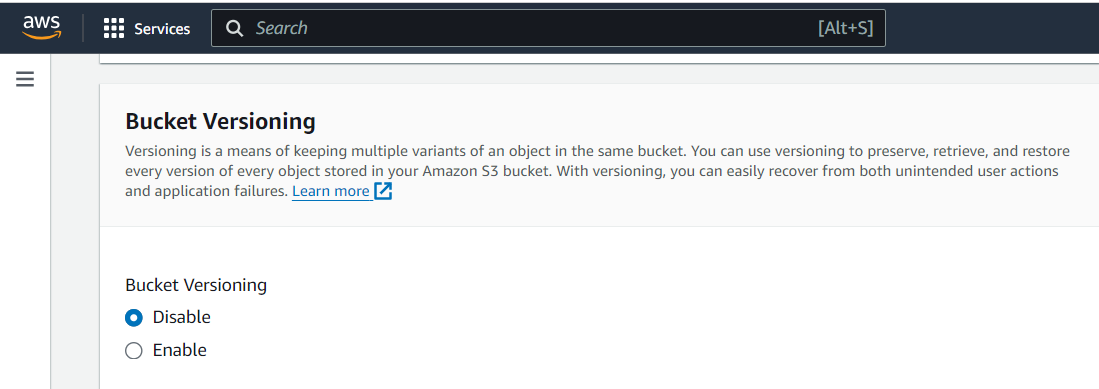
Create a S3 bucket, with no public access and upload files to the bucket & view the logs for the uploaded files. Launch two ec2-instances and connect it to a application load balancer, where the output traffic from the server must be an load balancer IP address

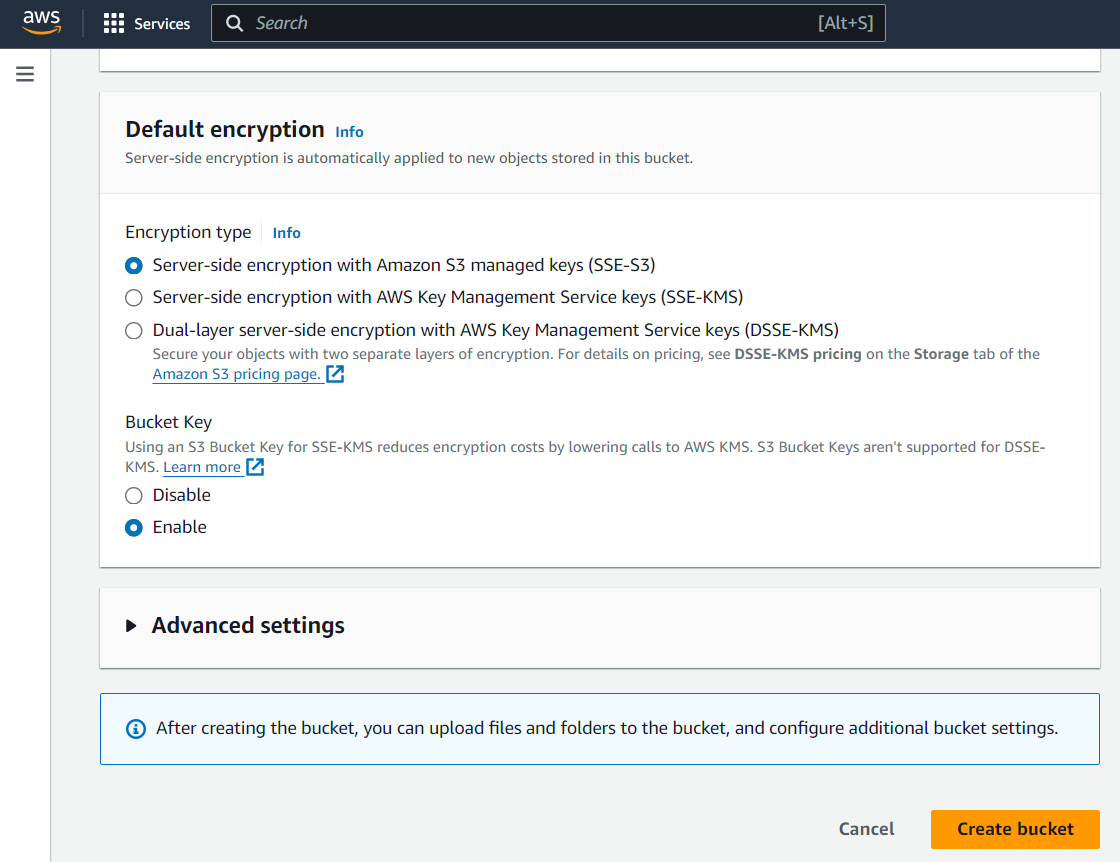
**Step 1: Create an S3 bucket**

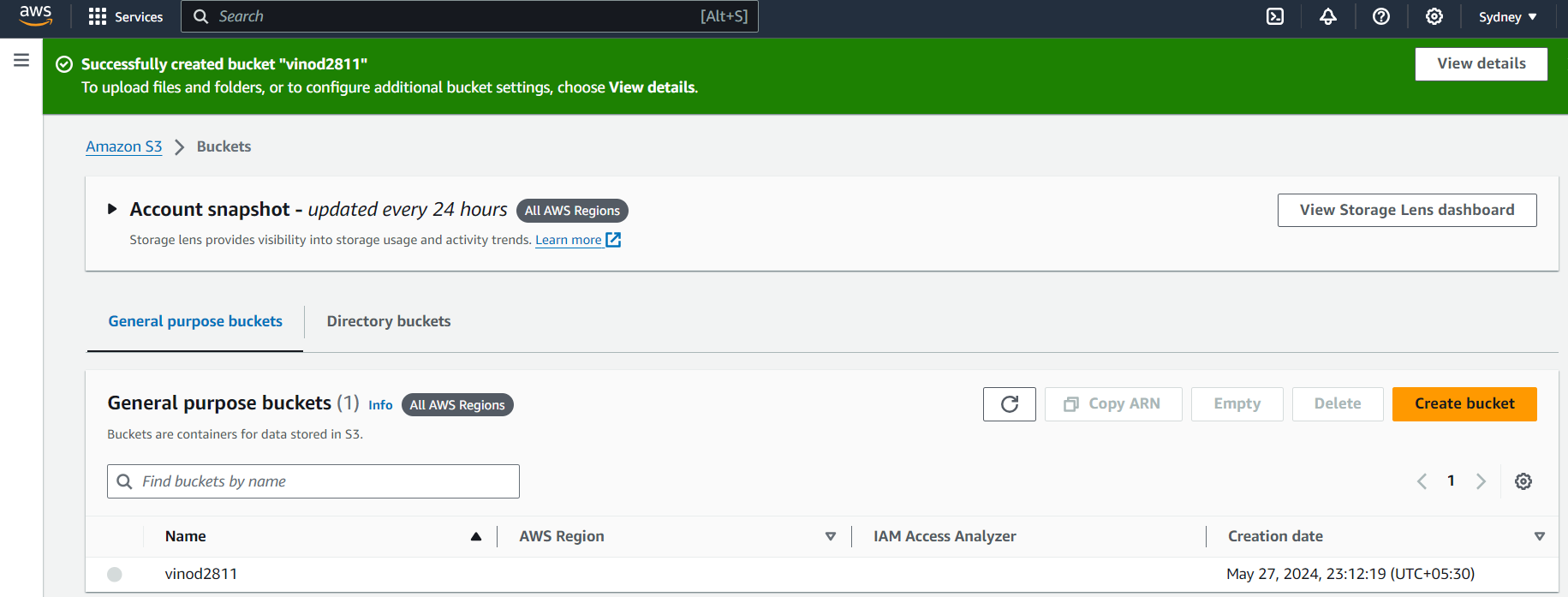


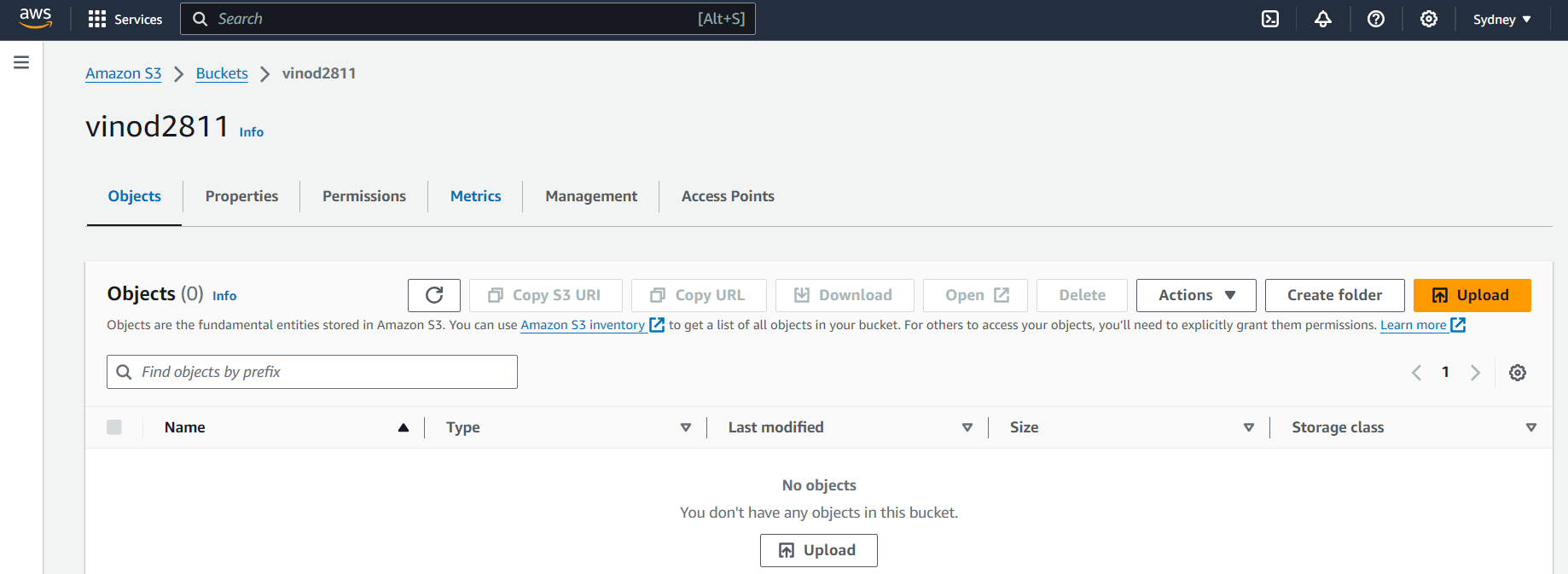




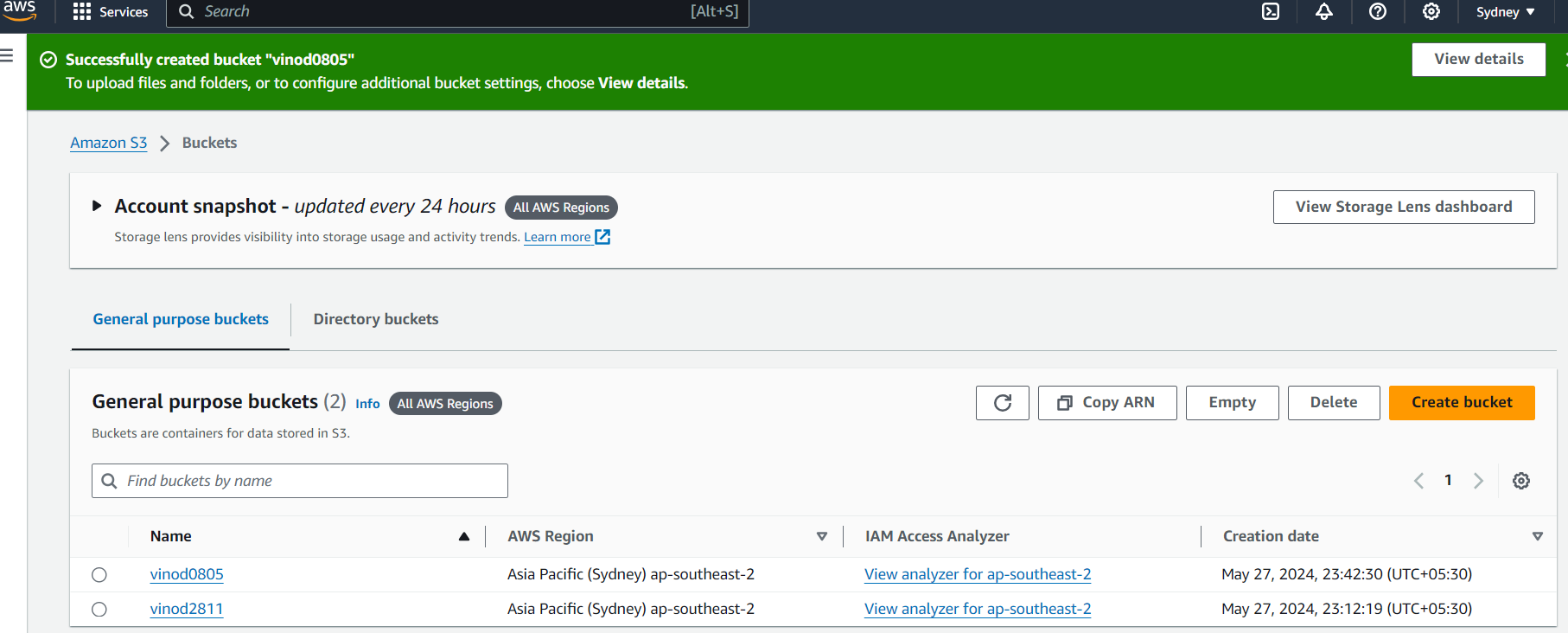




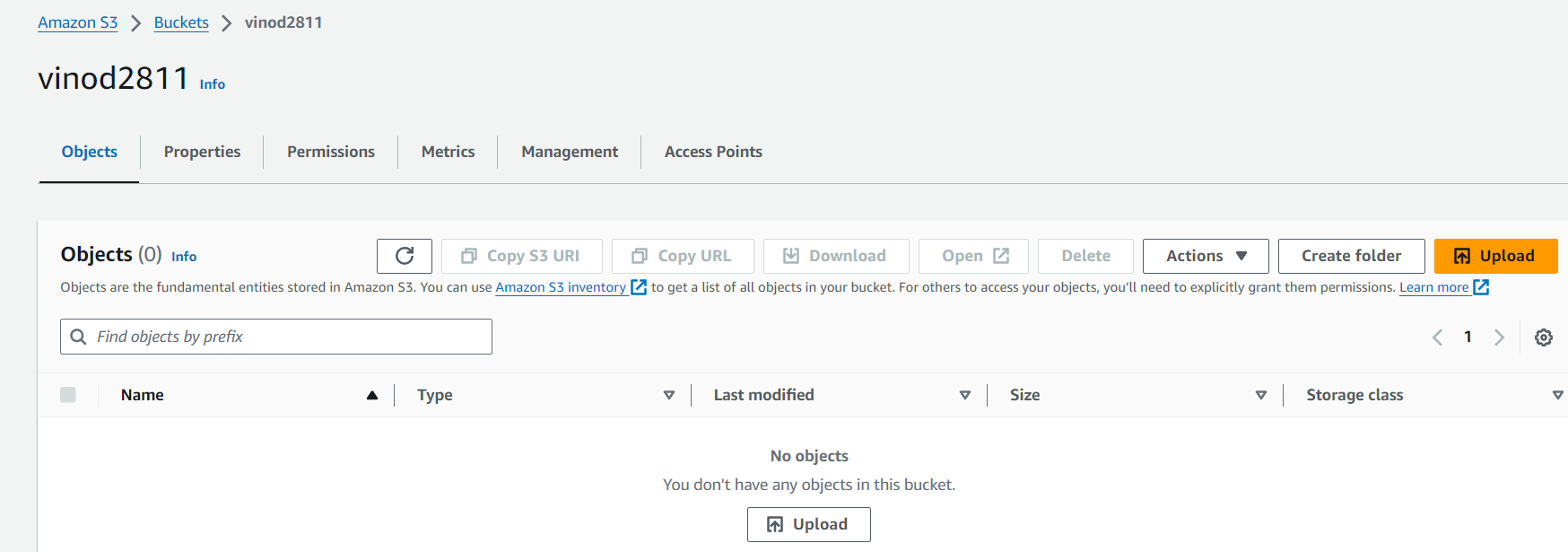


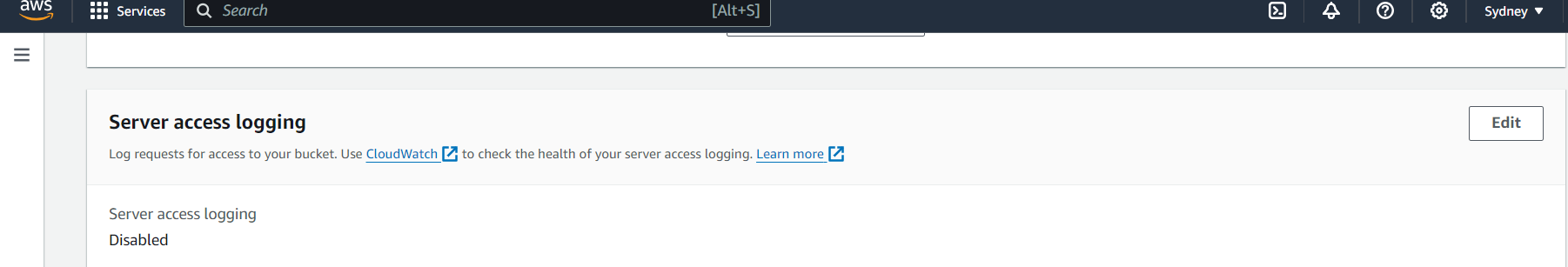


Let’s create destination bucket(for log monitor) by following similar steps used for source bucket

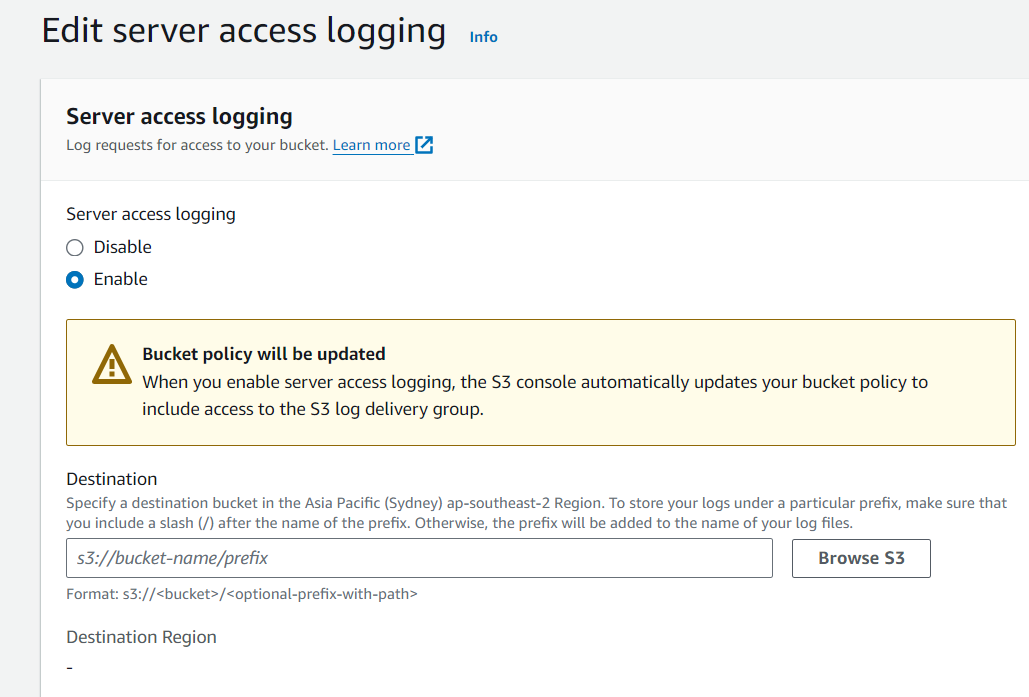


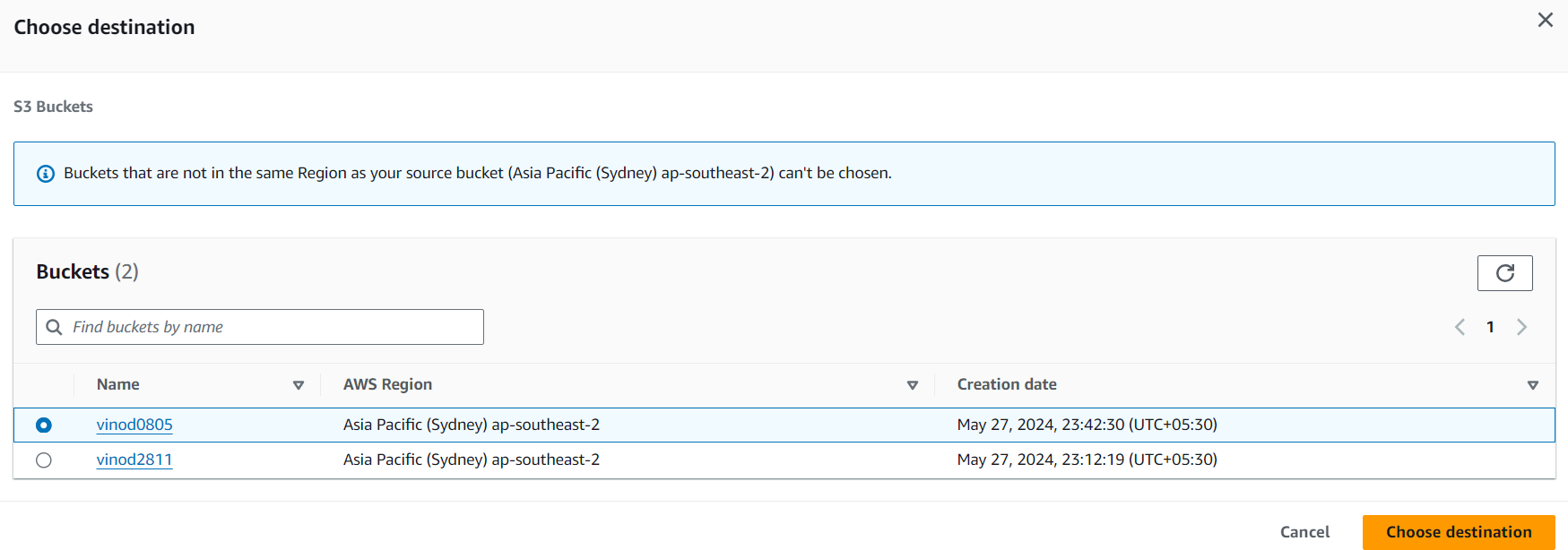
**Goto -> Properties section of source bucket vinod2811 -> To enable logs**

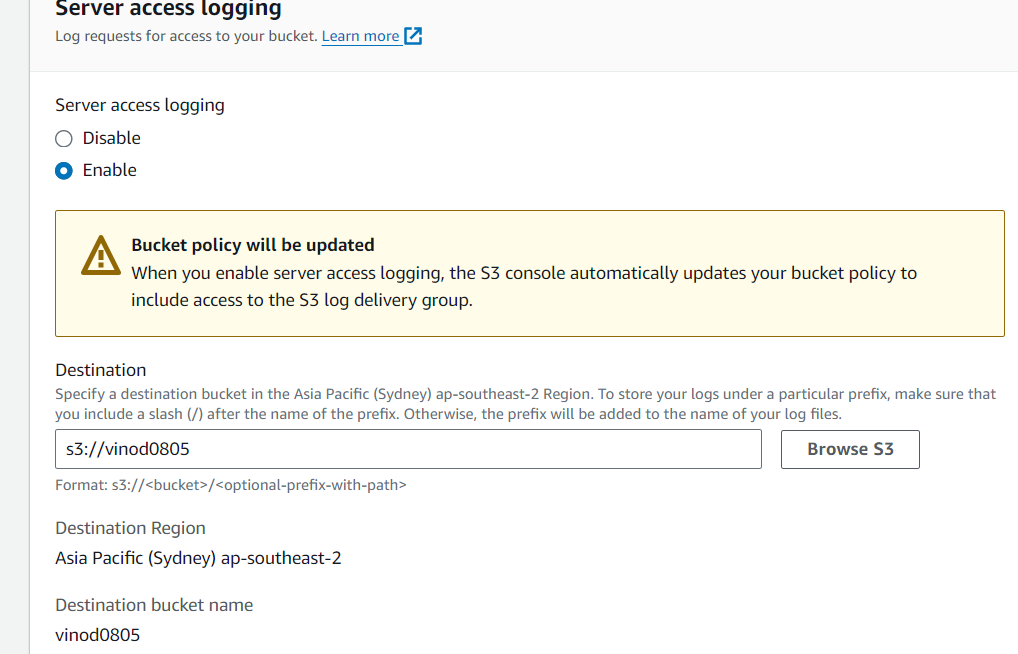


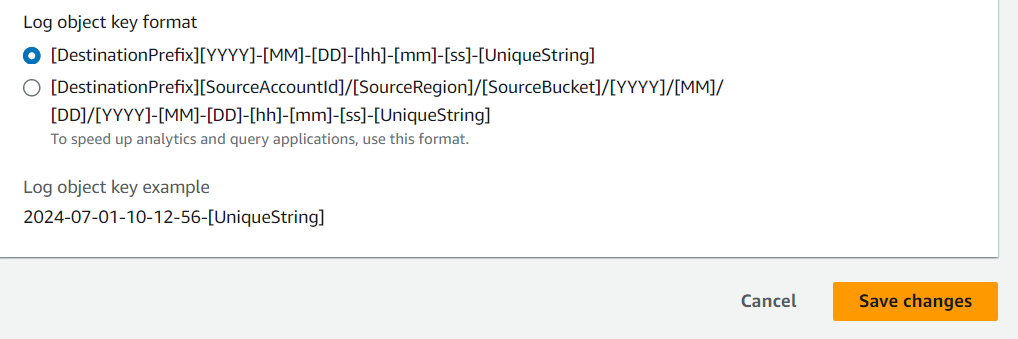


**Enable the server access logging and choose respective bucket for log redirection**

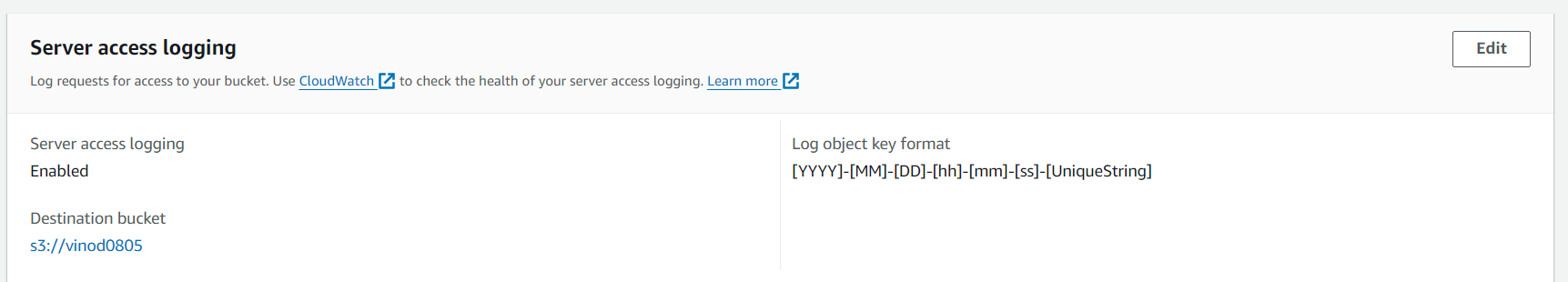




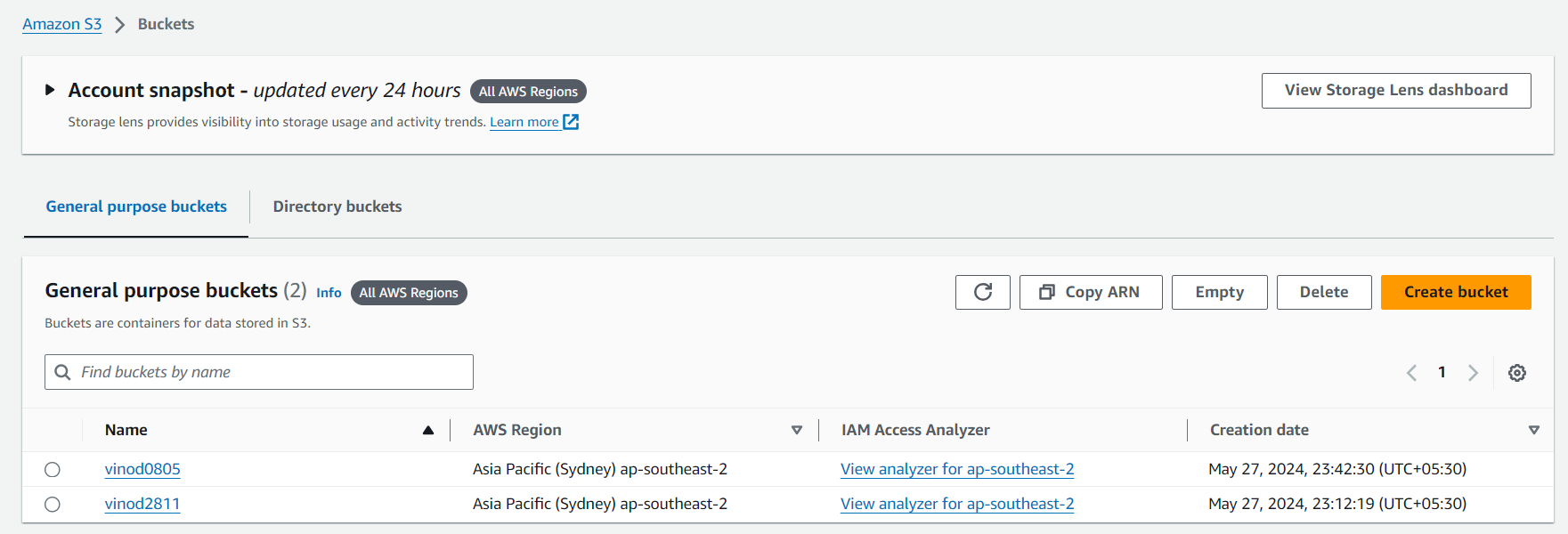


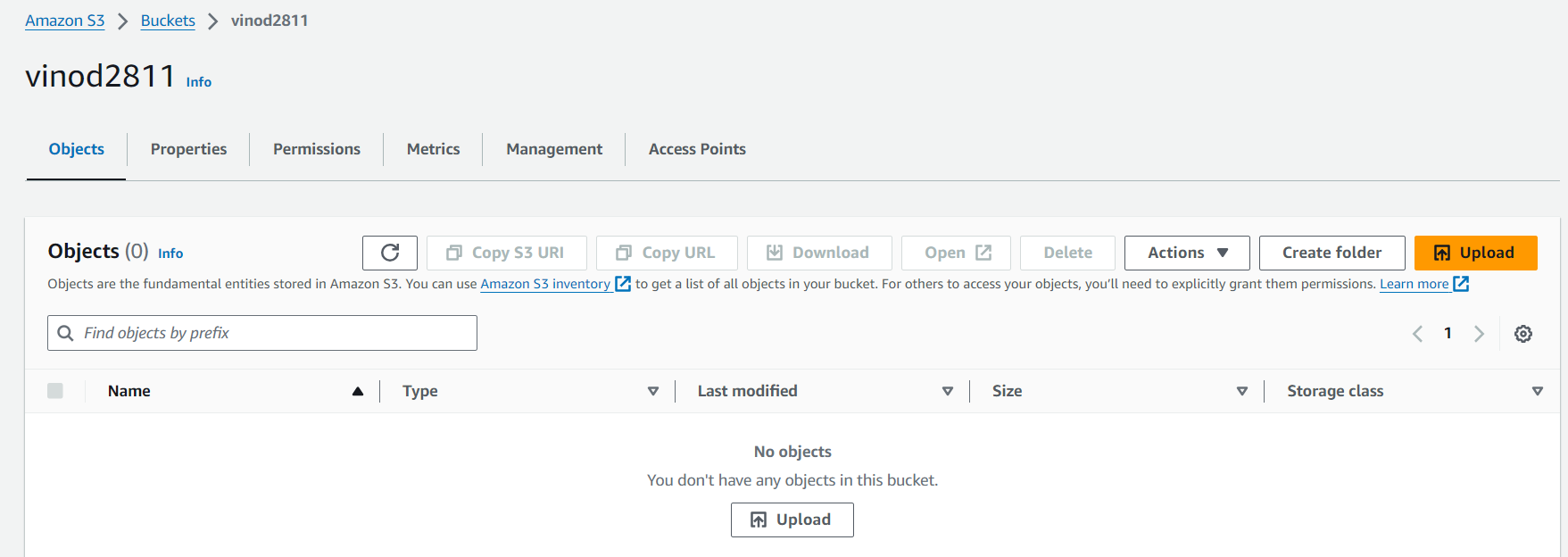


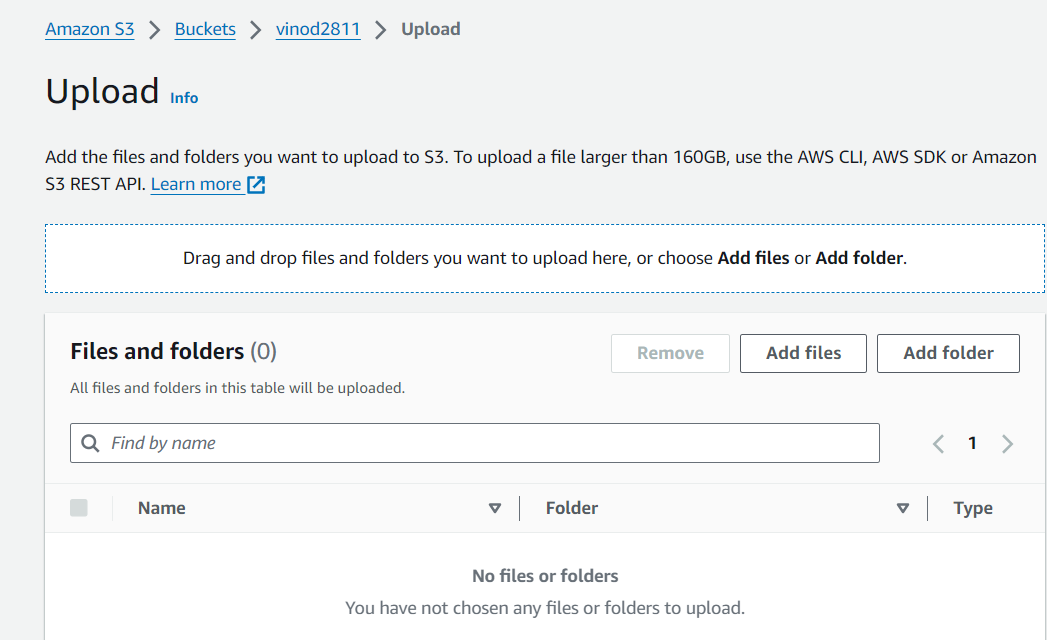
**Verify the sever access log enabling**

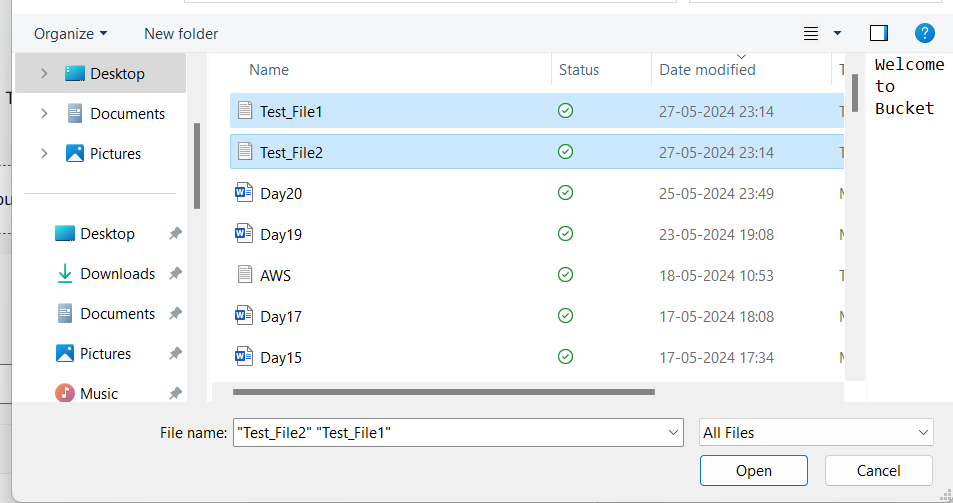


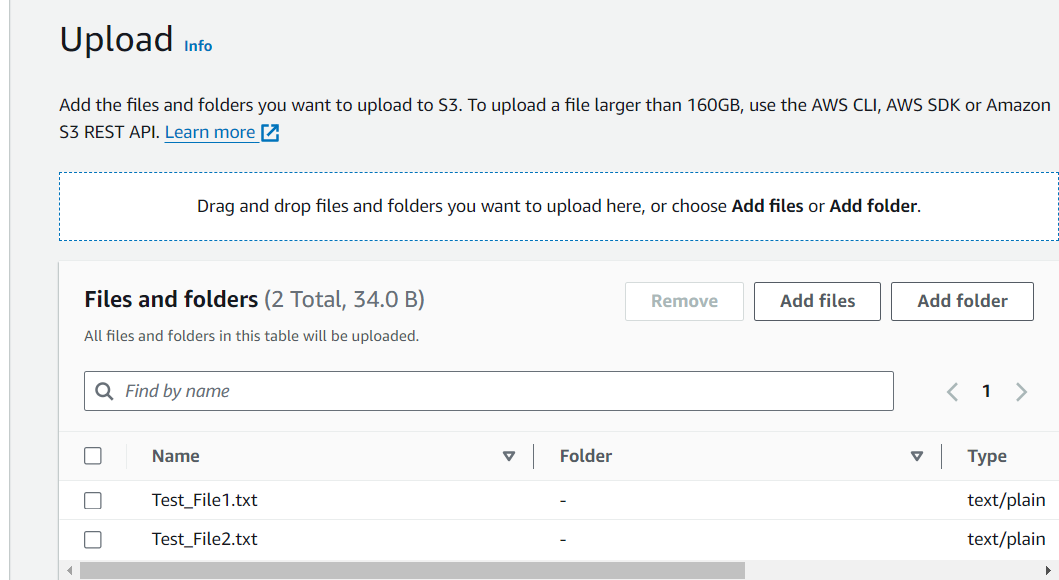
**Now we can try uploading some files to the source bucket ( vinod2811)**

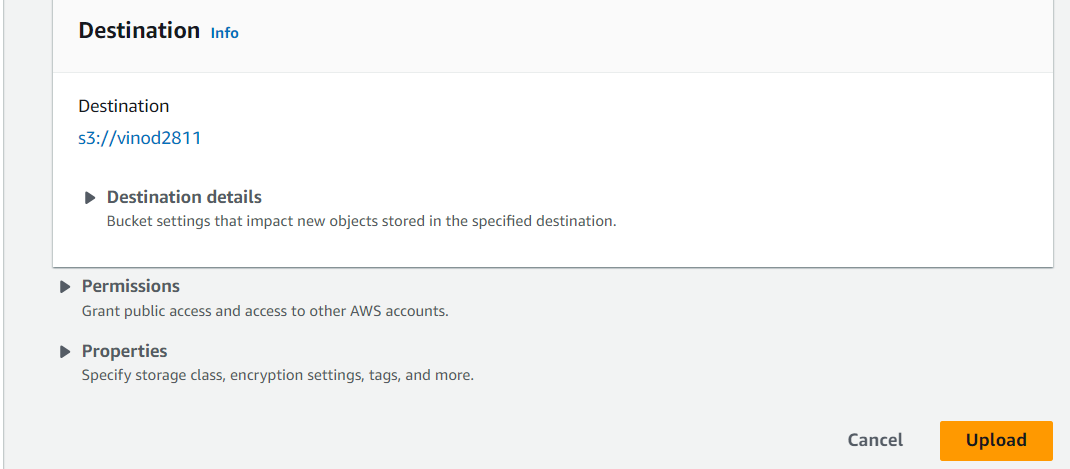


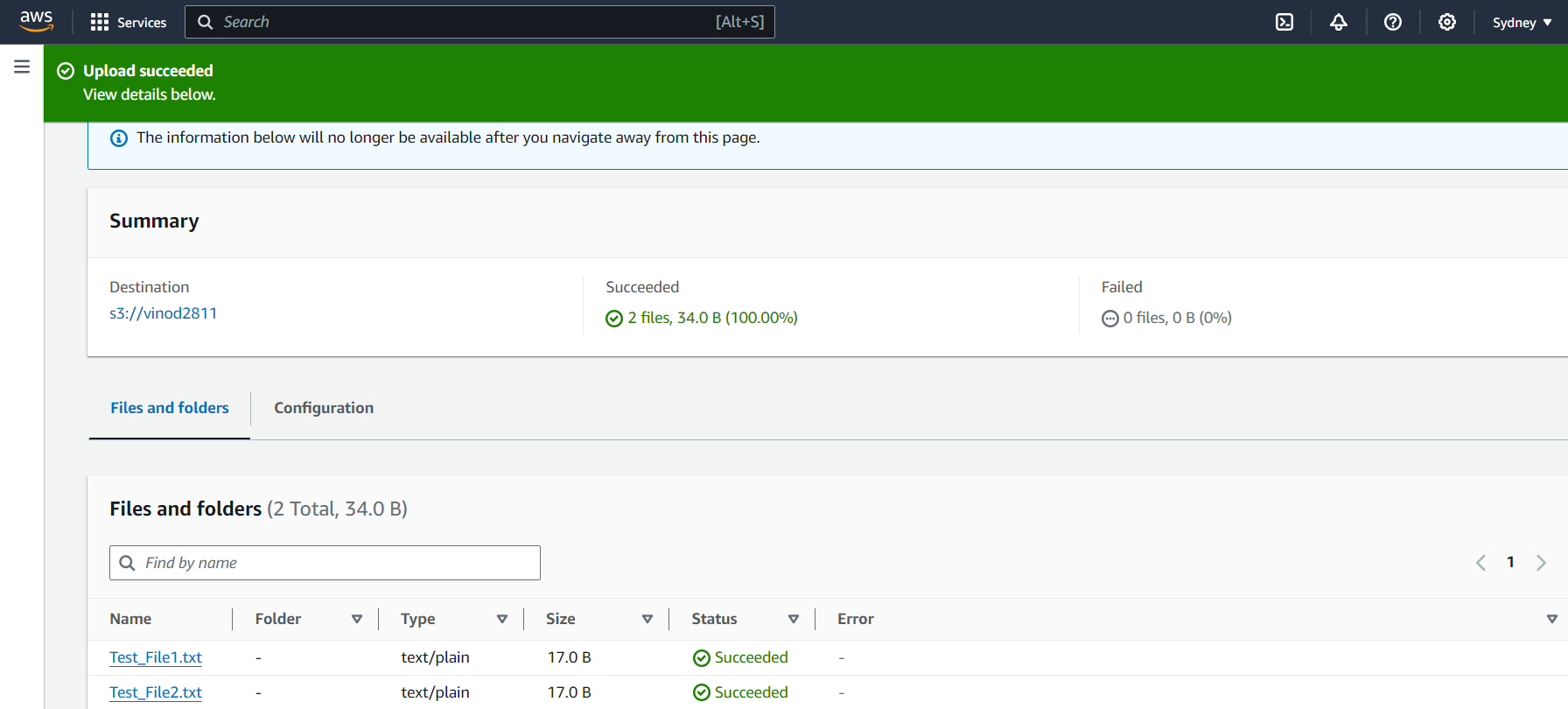




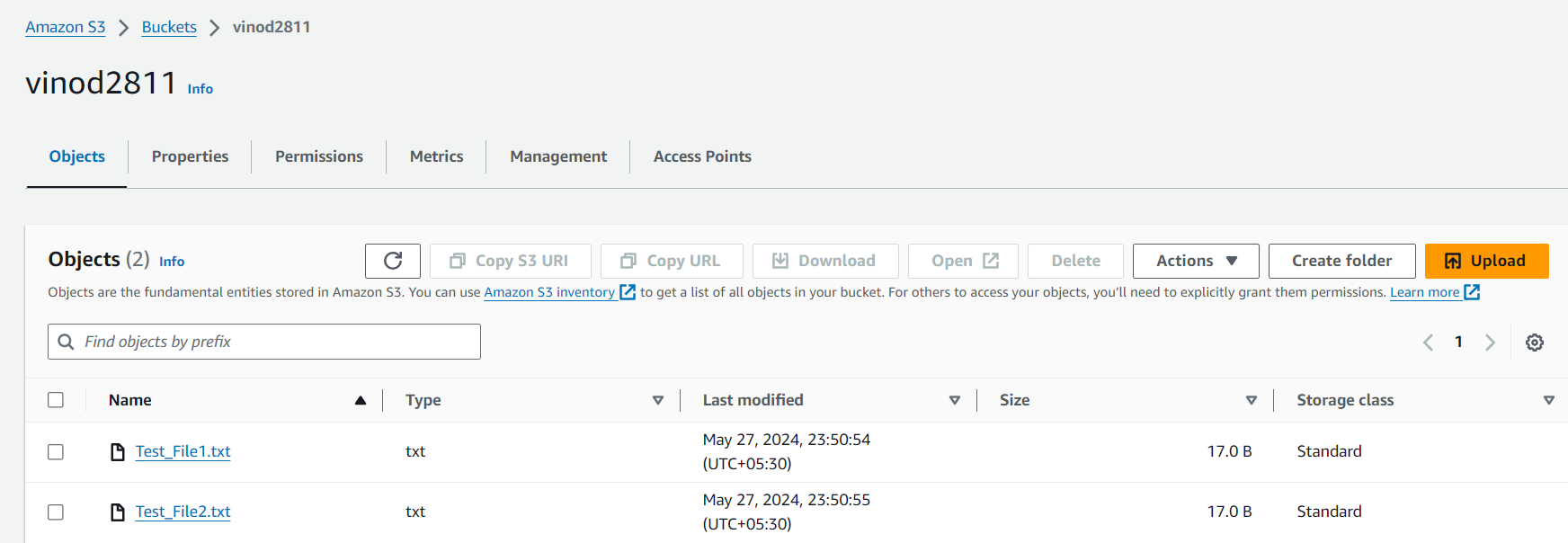






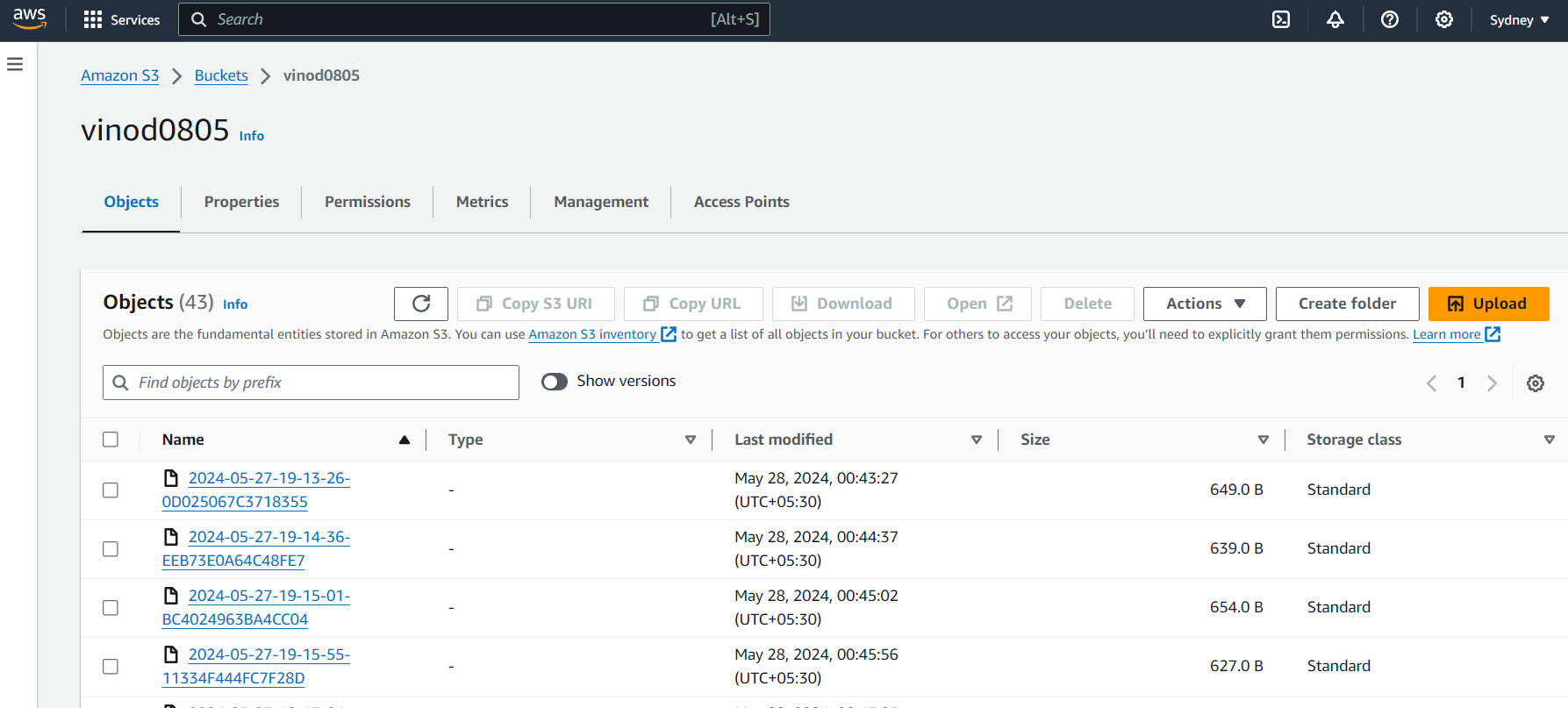


**Files uploaded :**



**Logs Verification:**

We can view the server logs for the uploaded files on below destination path ( vinod0805)

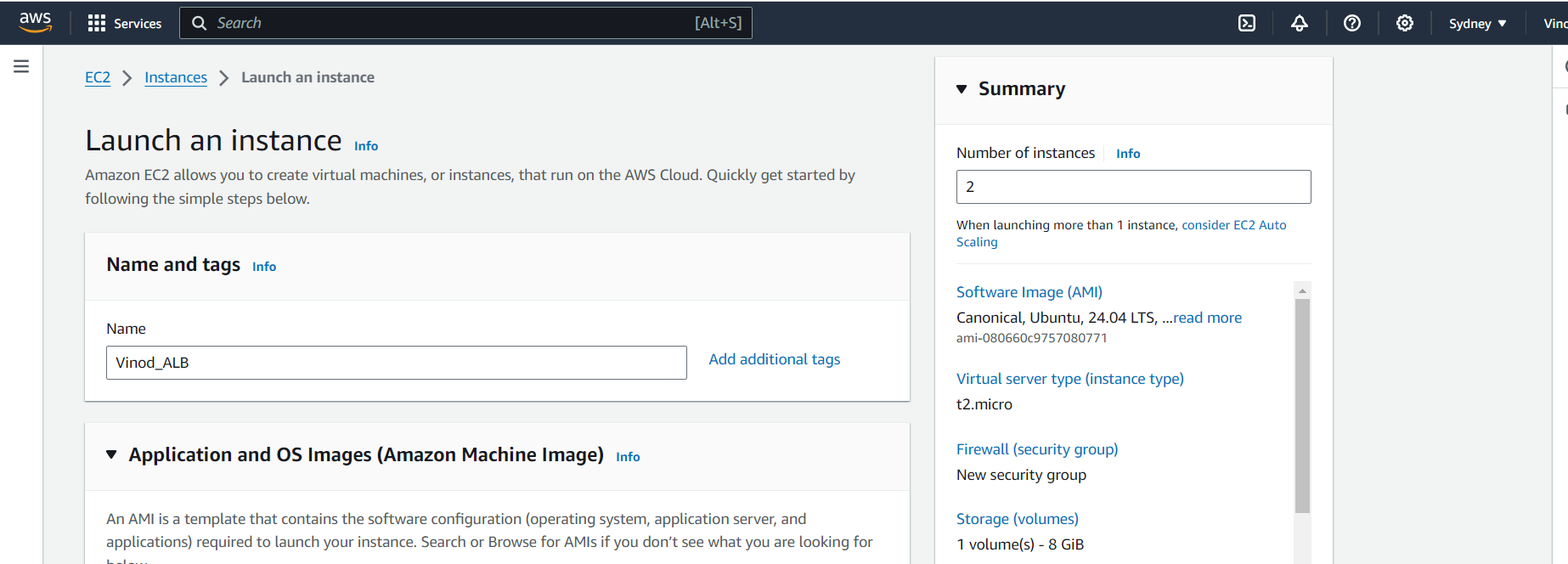


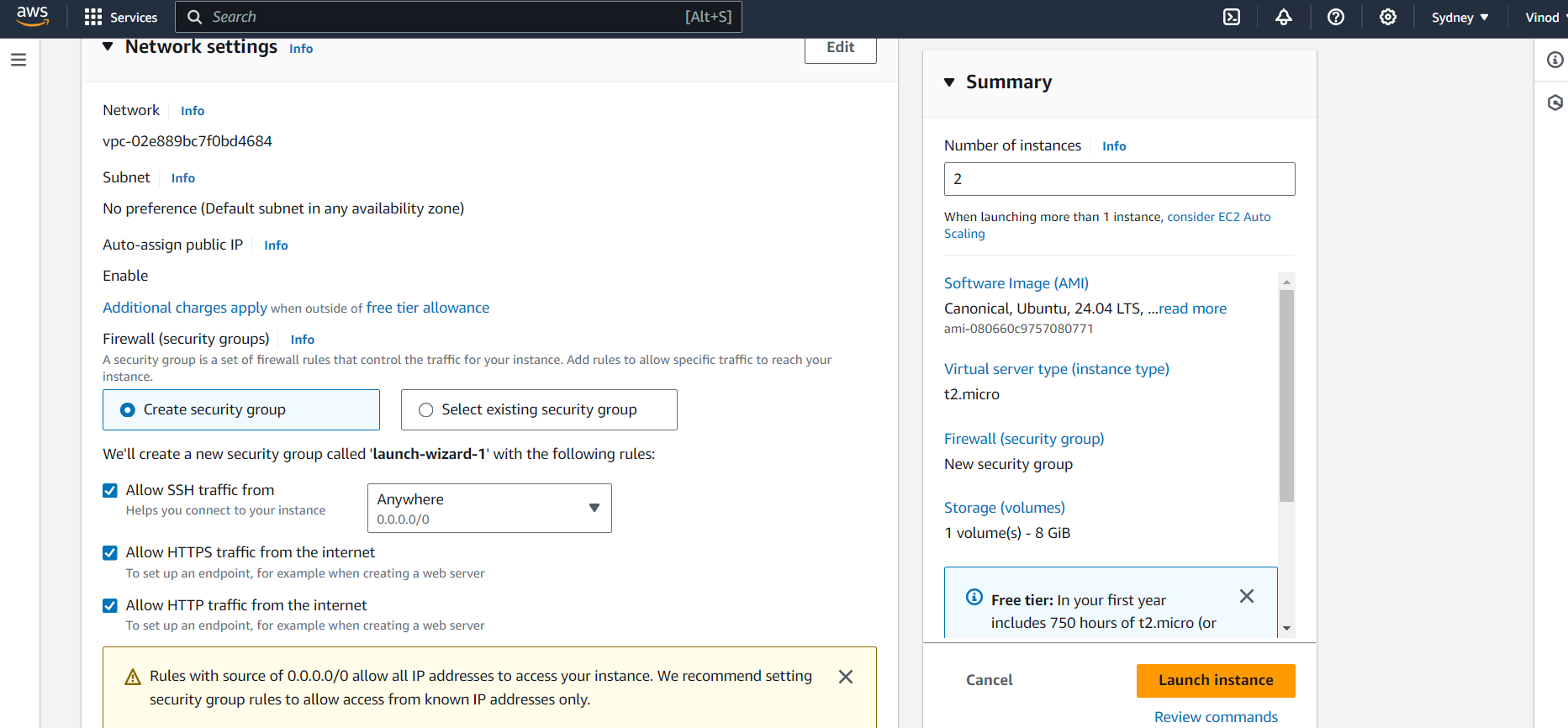
Hence files are uploaded and logs are verified on S3 bucket accordingly

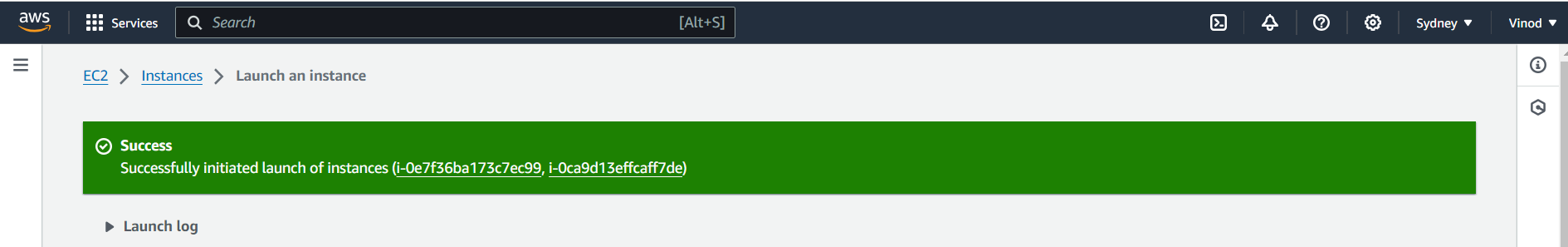
As a good practise of learning, post validation we can remove the buckets to avoid unwanted charges

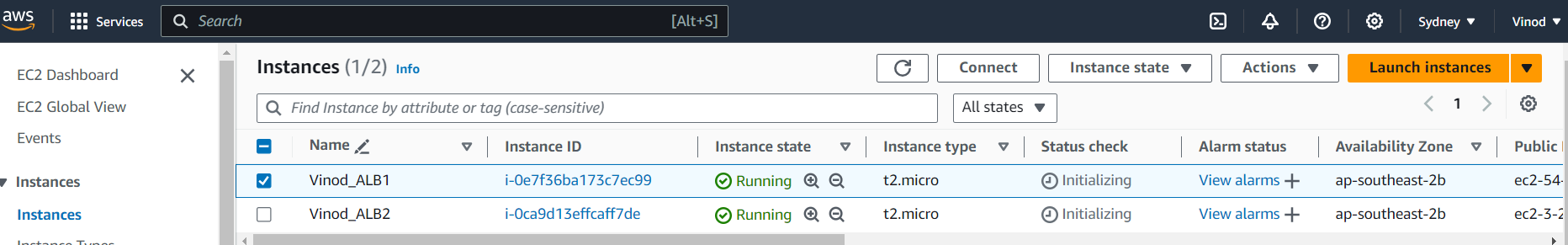
**Note** : It’s not recommended to use same bucket for source and destination (for log capture) it could result in a small increase in your storage billing .Hence we used different one for logs

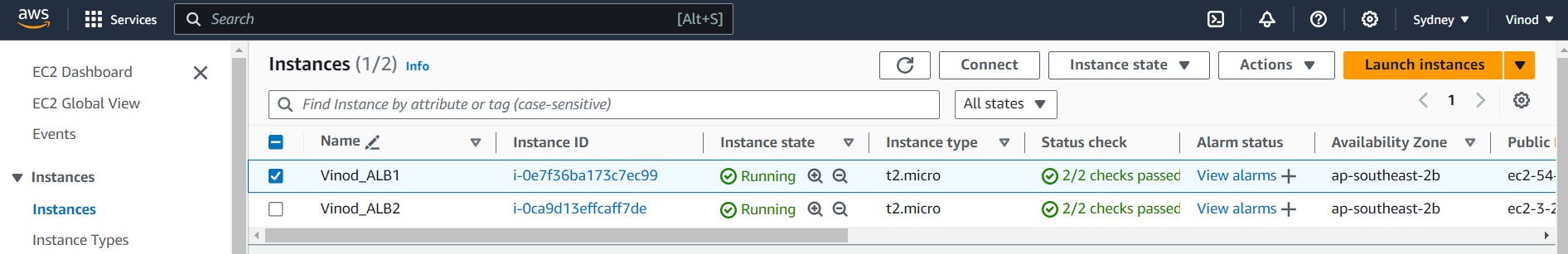
**2) Launch two EC2 instances**

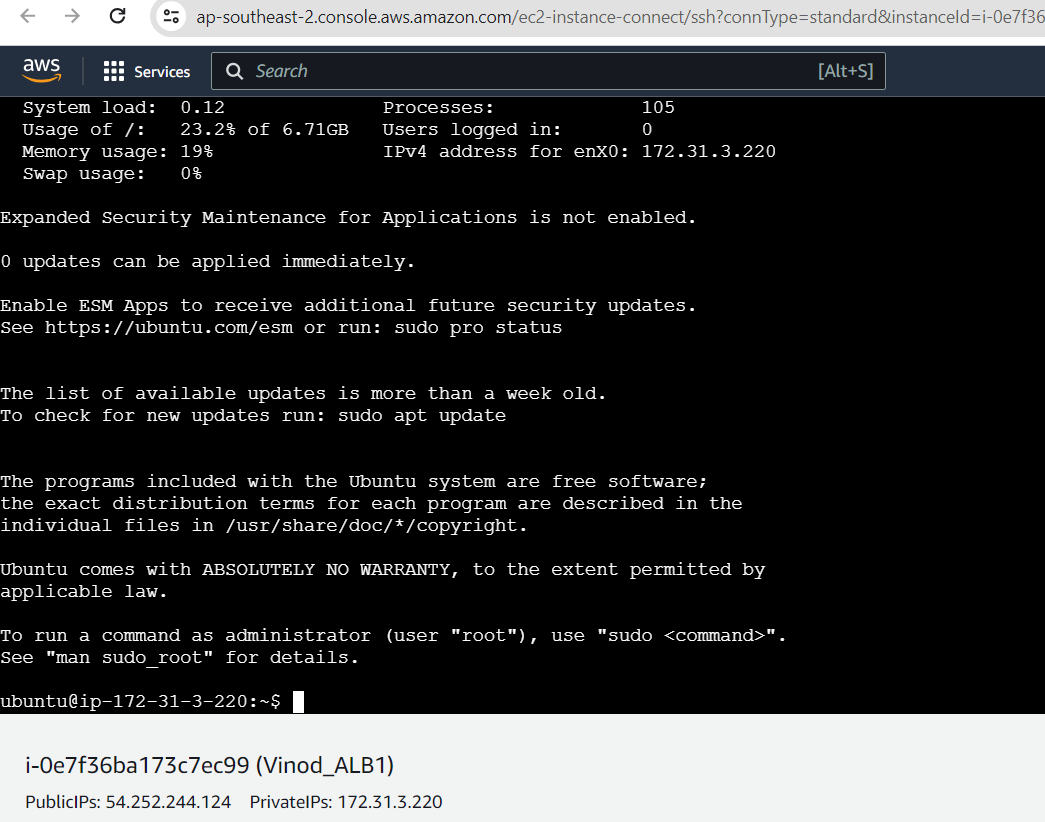


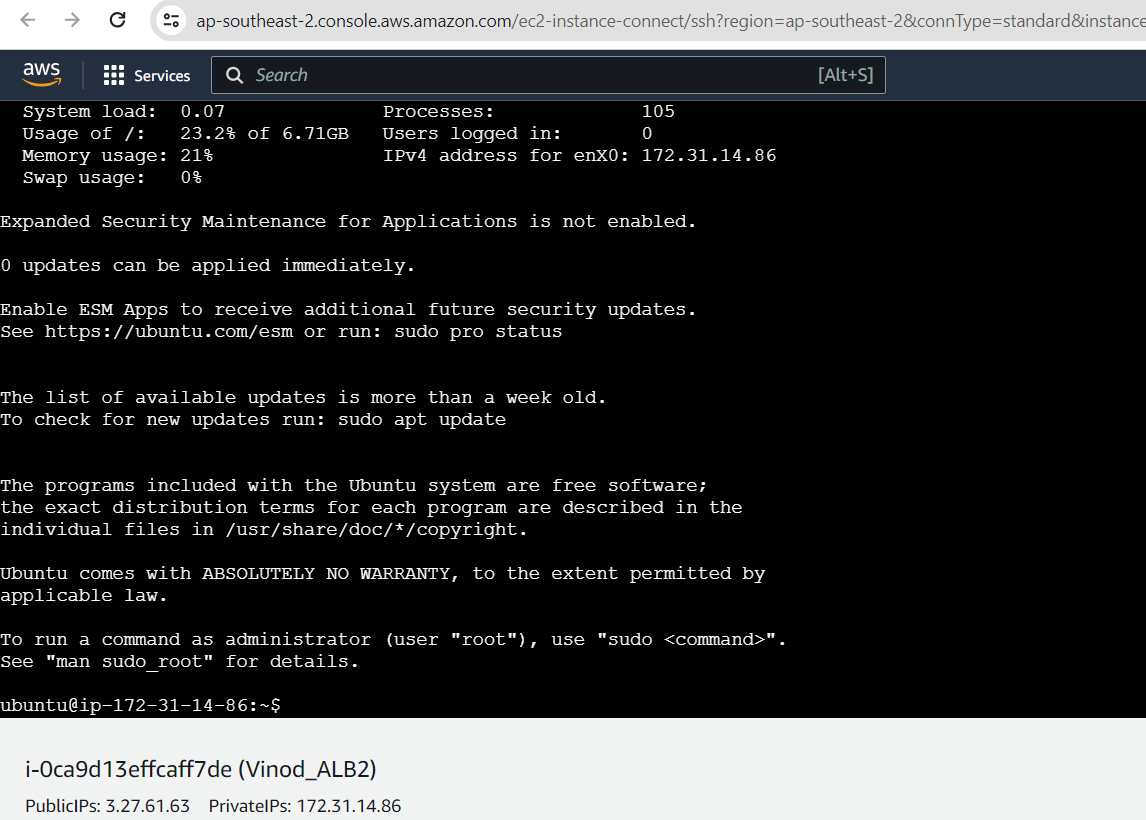




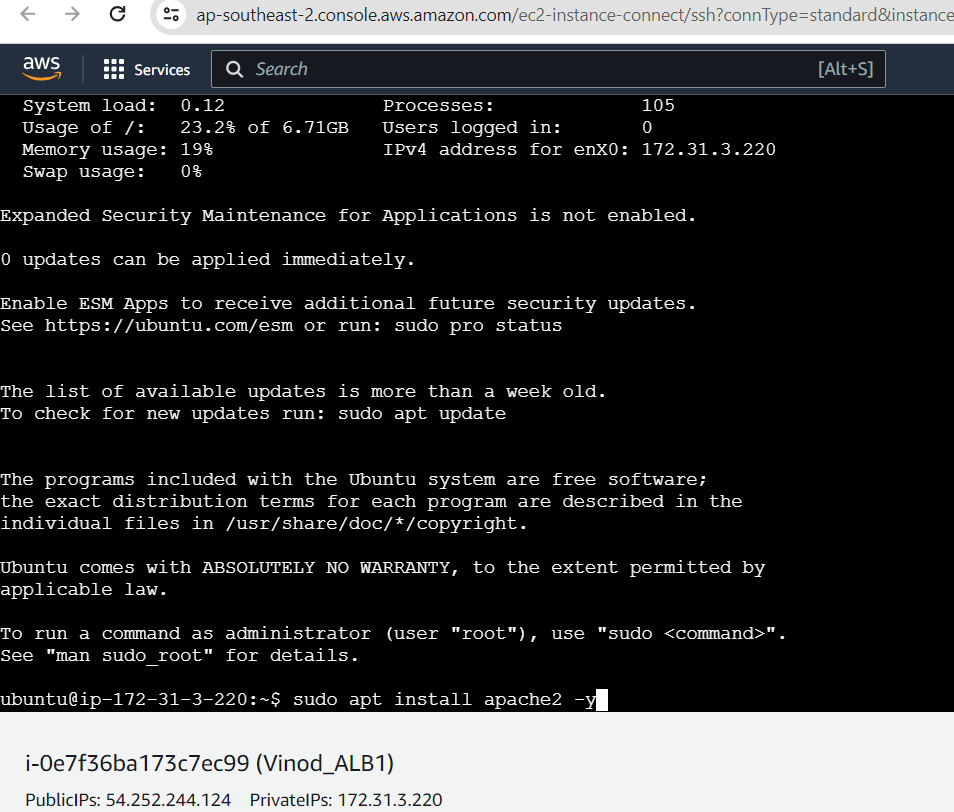


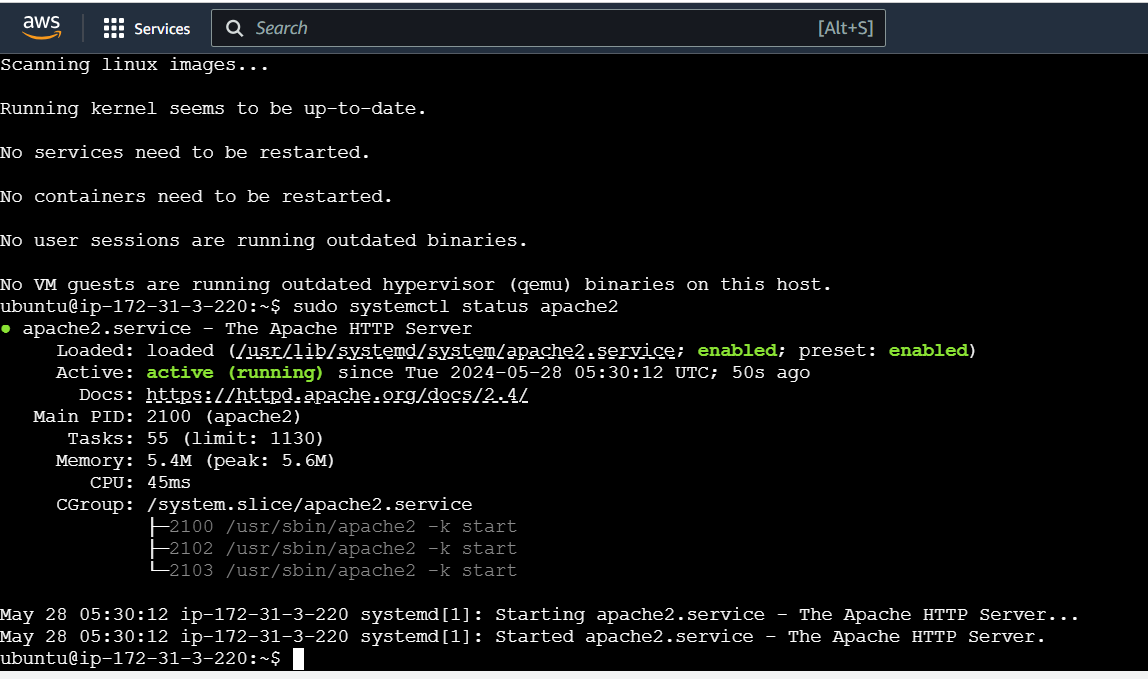






**Install apache on instance1**

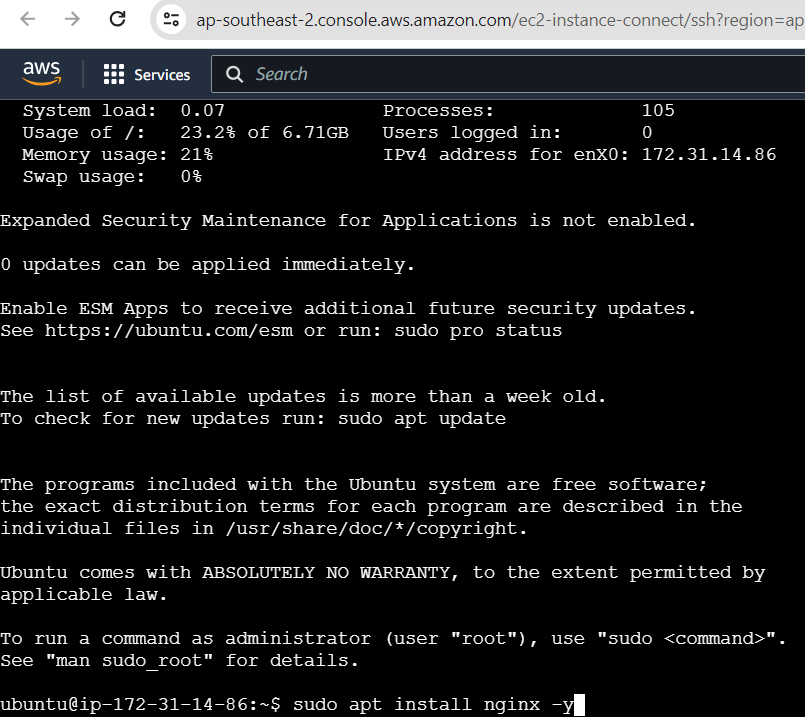


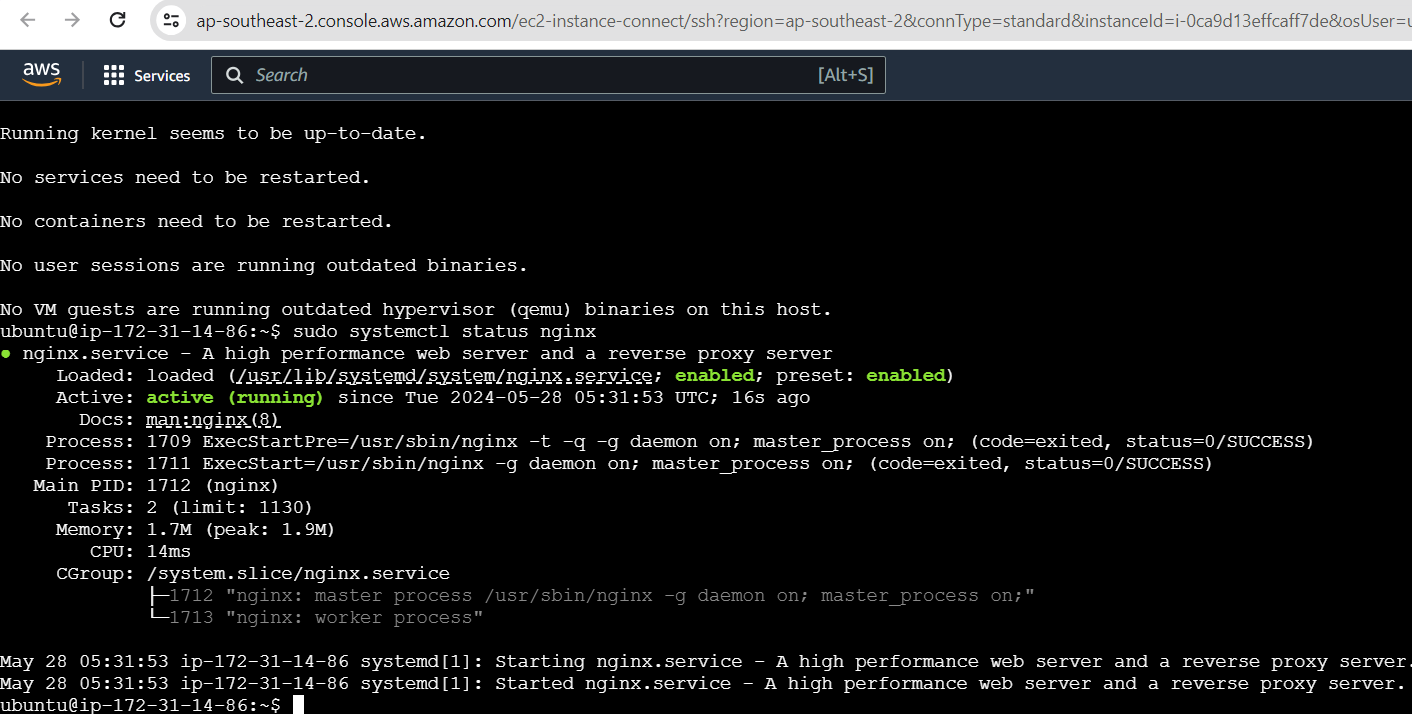


**Application running – Verified**

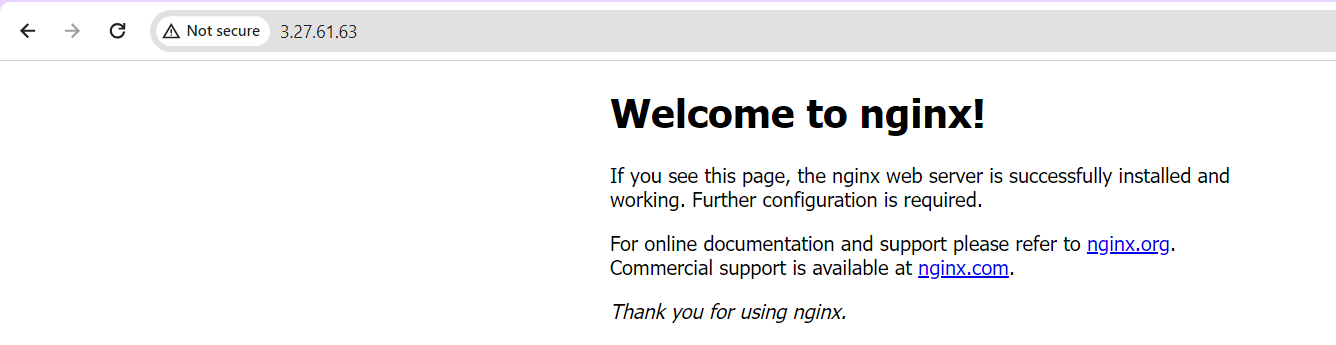


**2 ) Install nginx on instance 2**

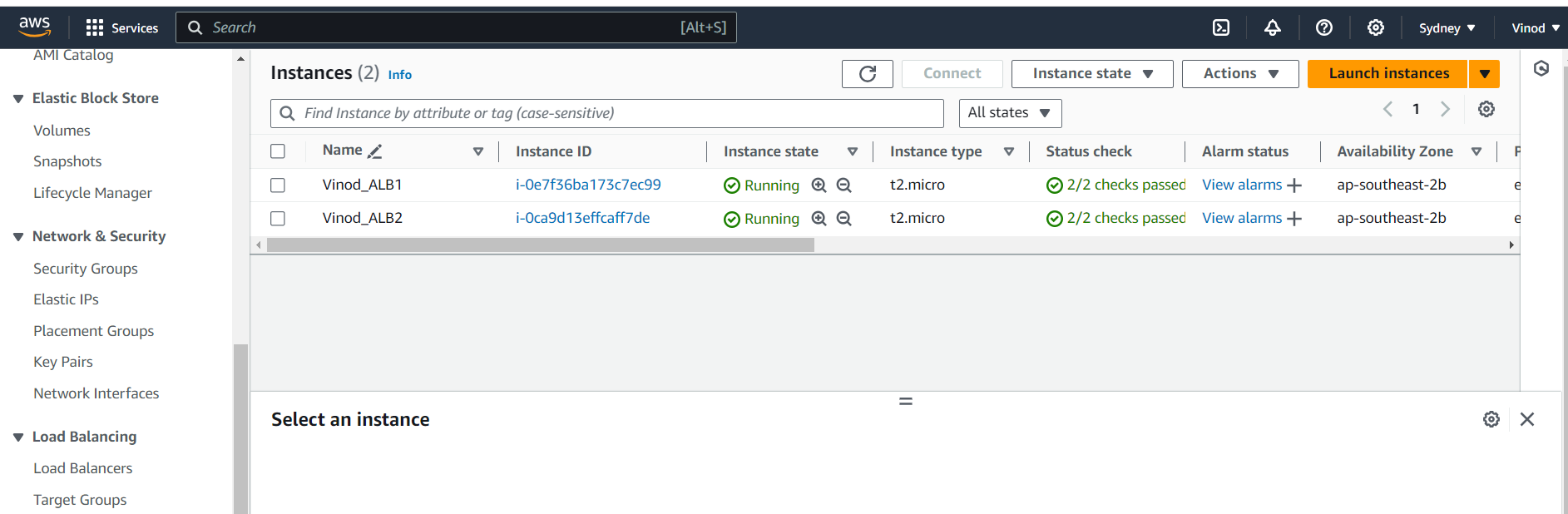


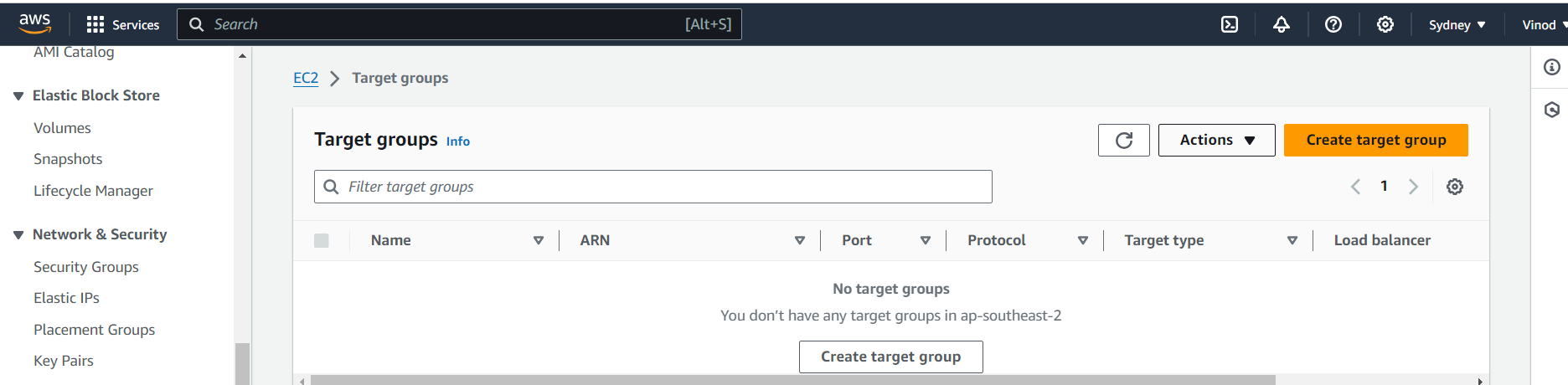


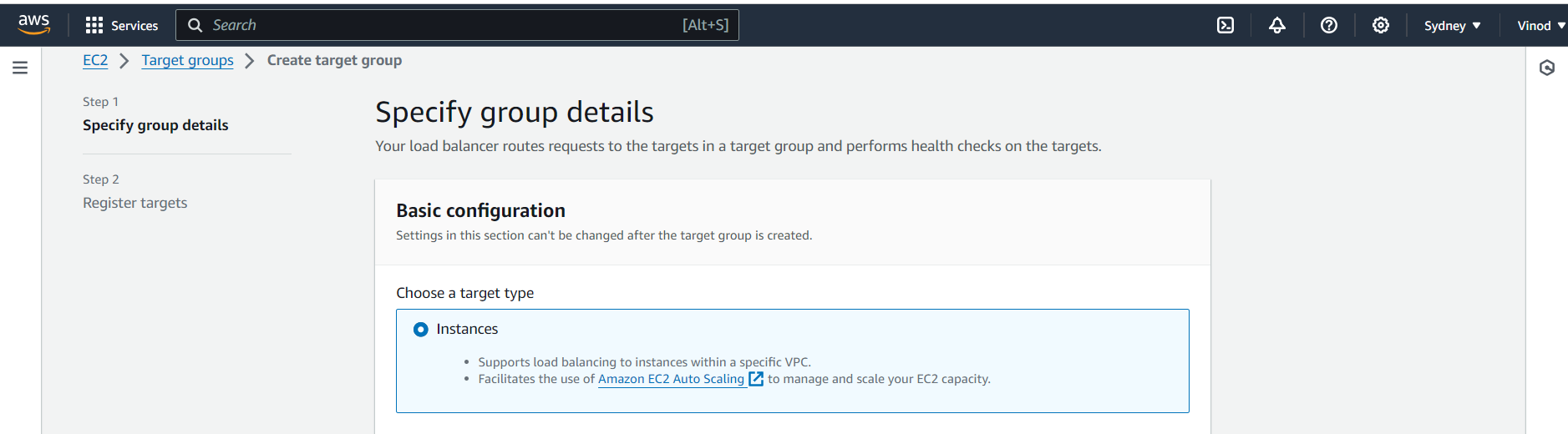
**Application running - Verified**

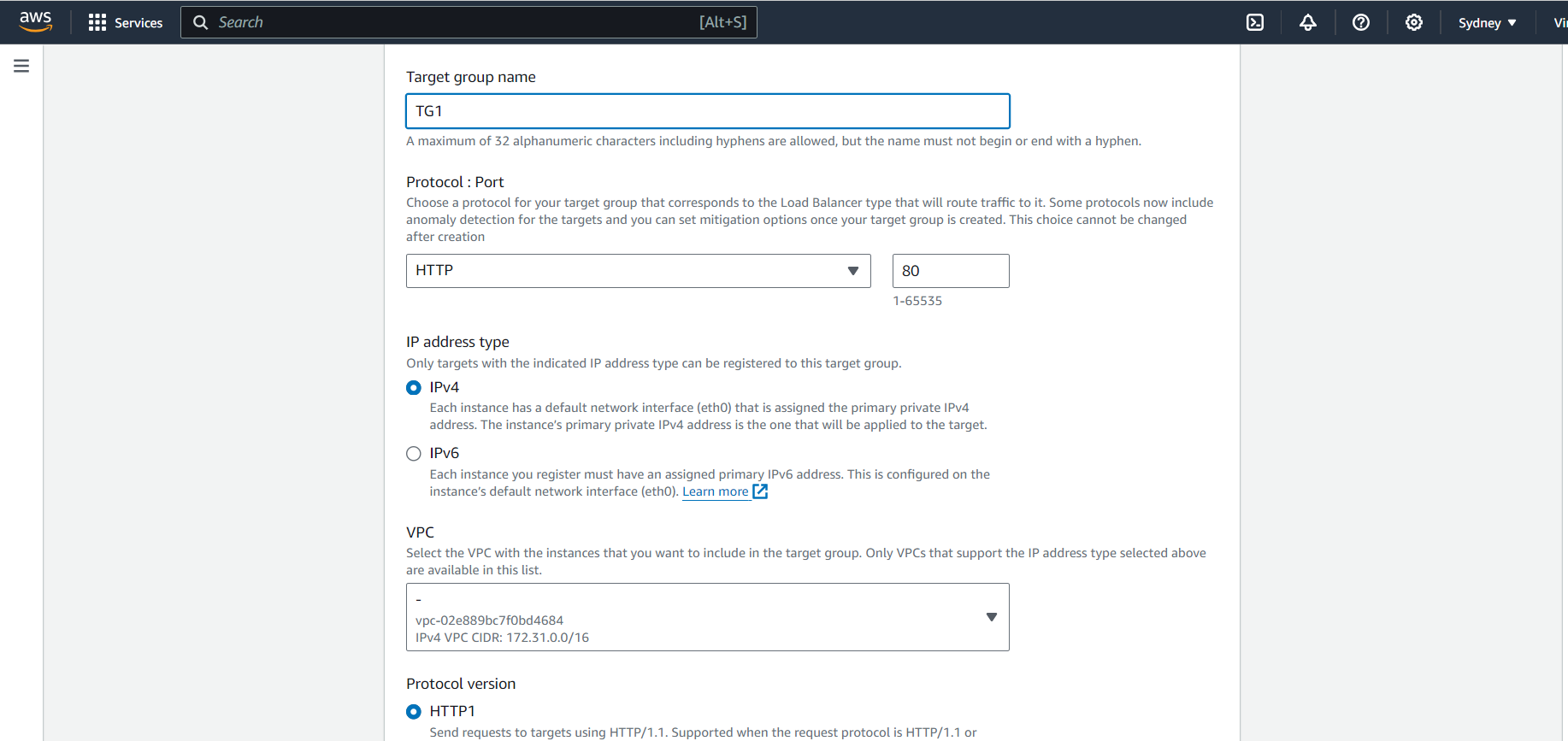


**Create Target Group :**

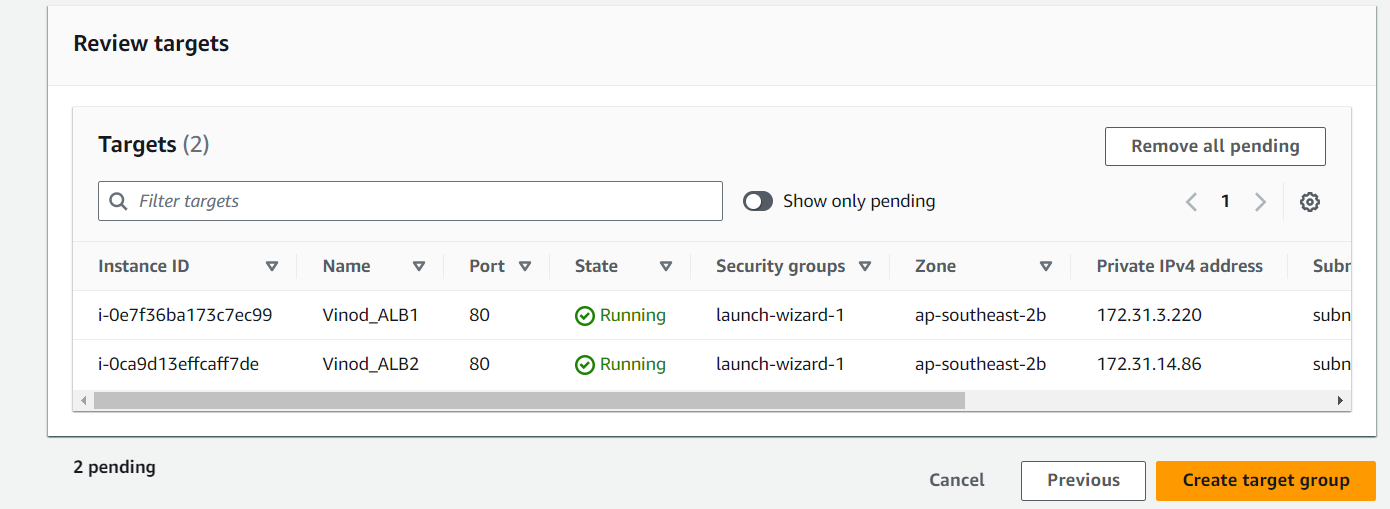


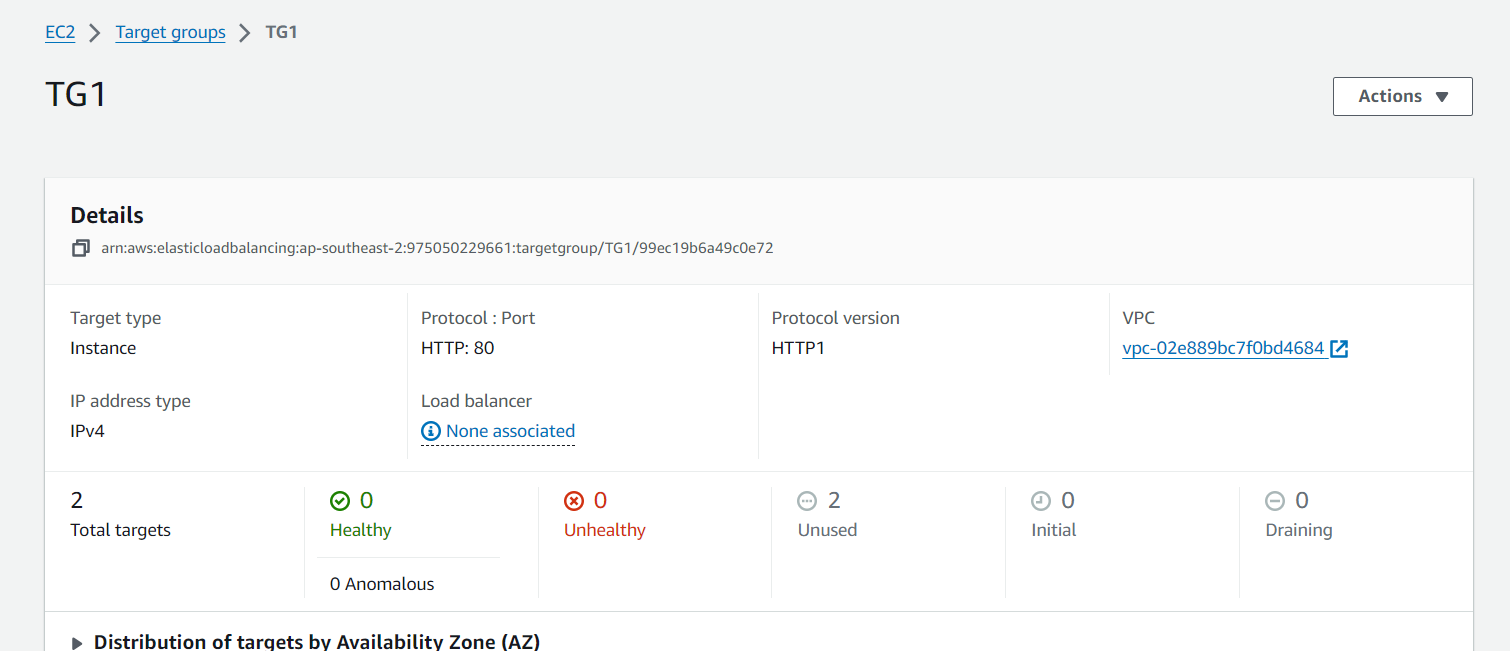


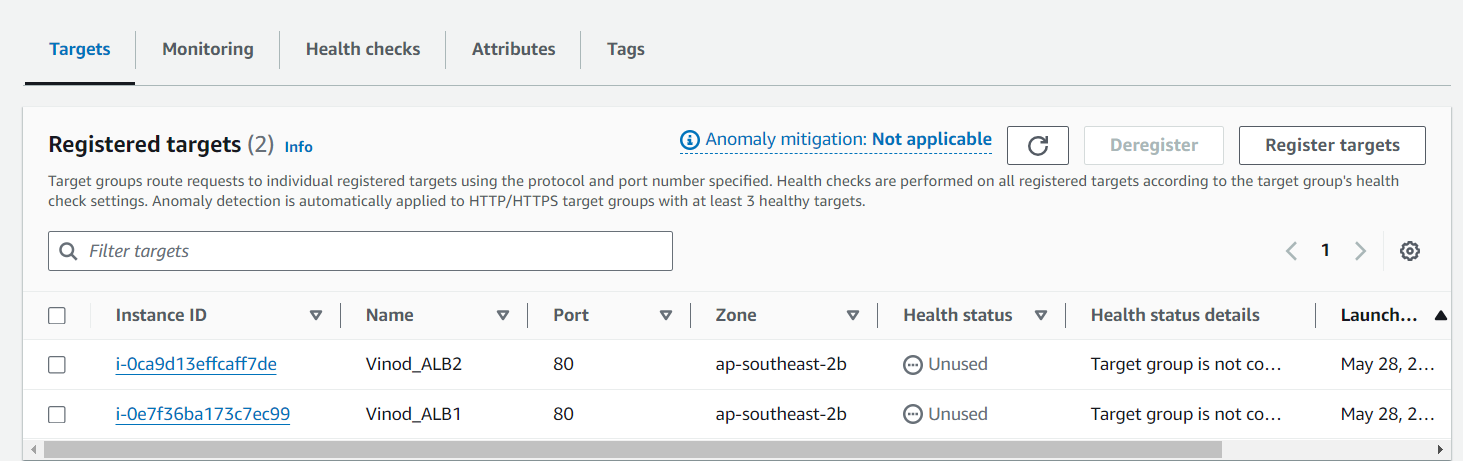




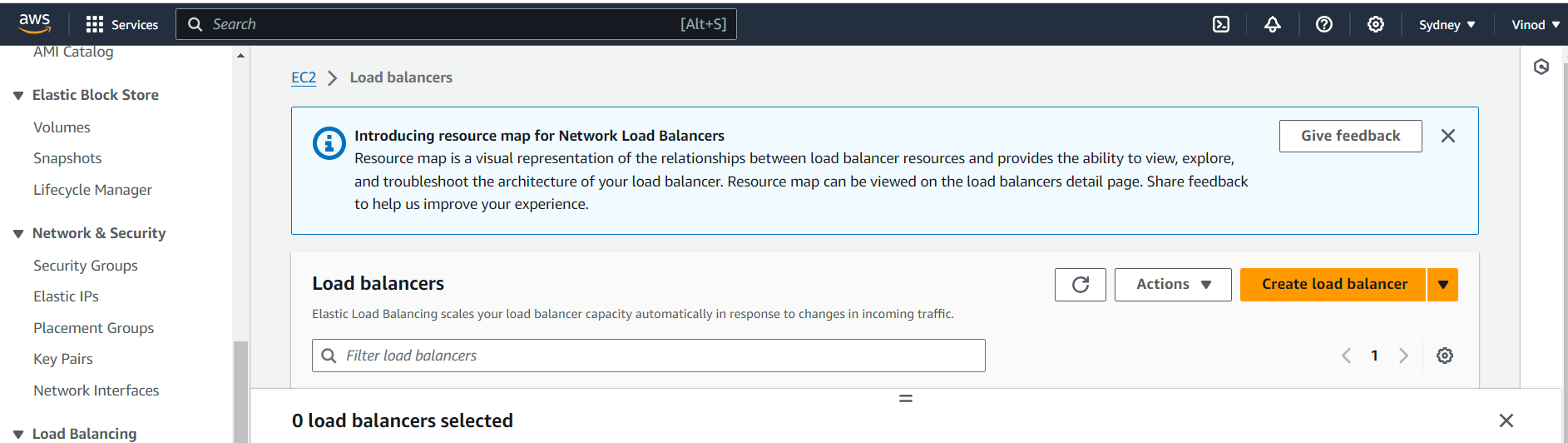


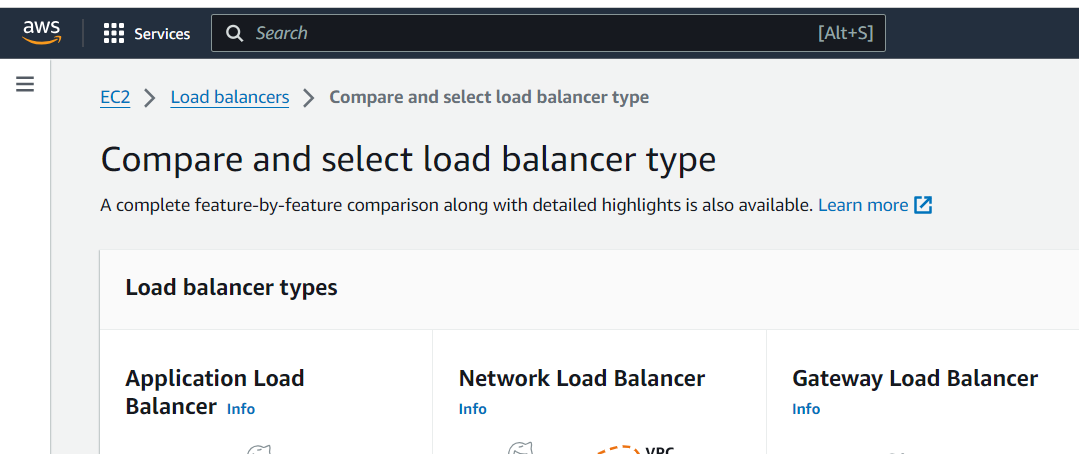


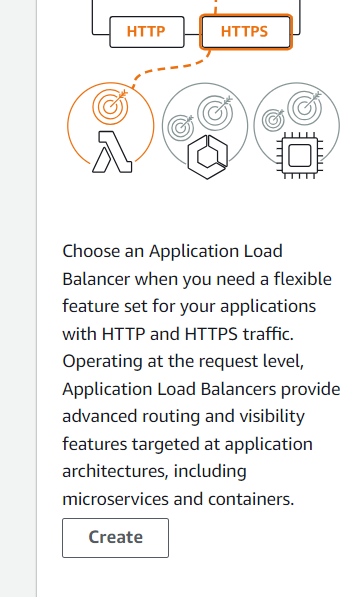


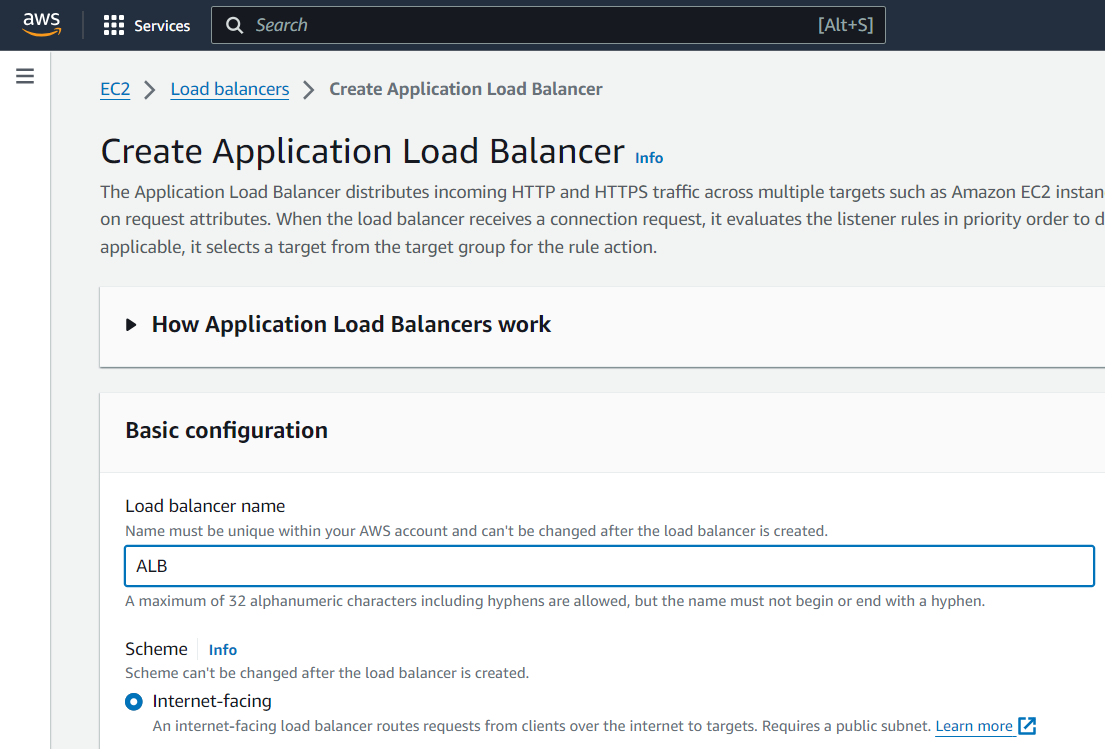


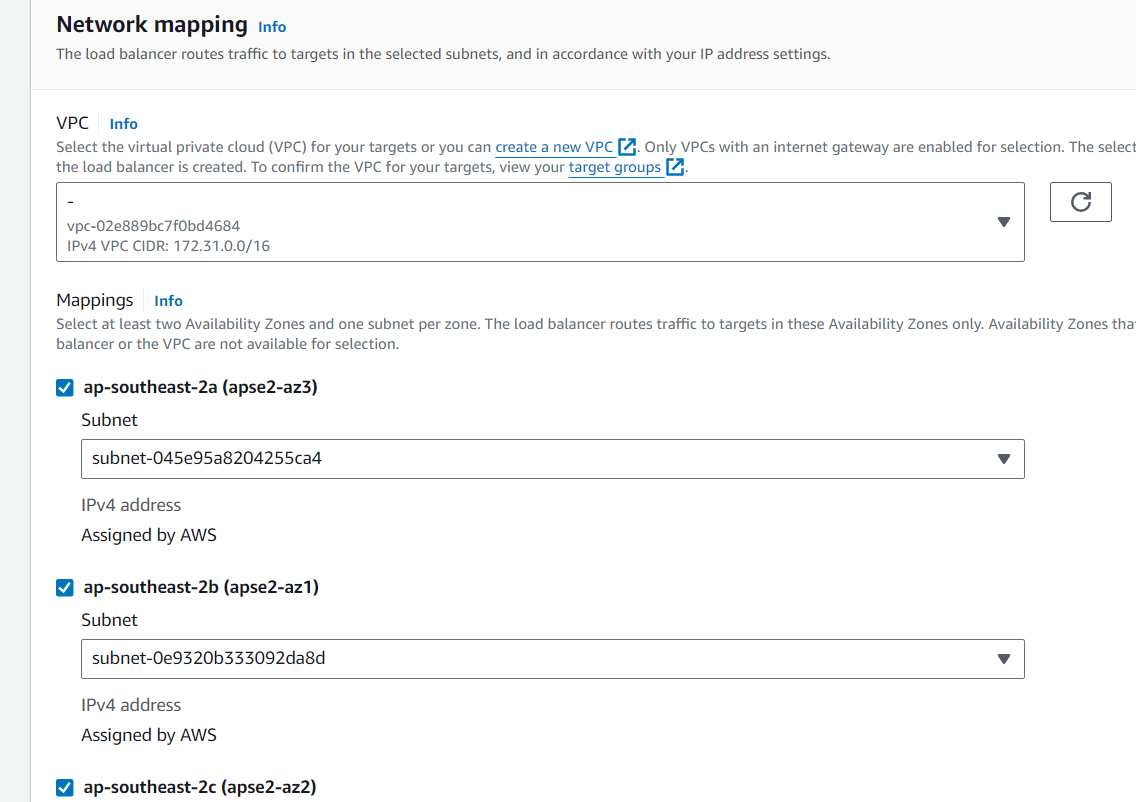
**Create Load Balancer:**

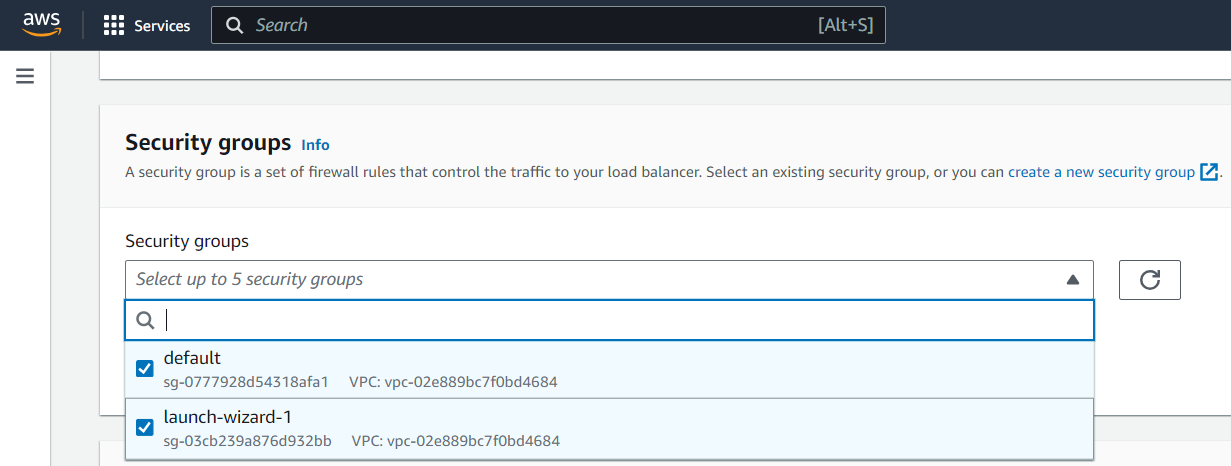


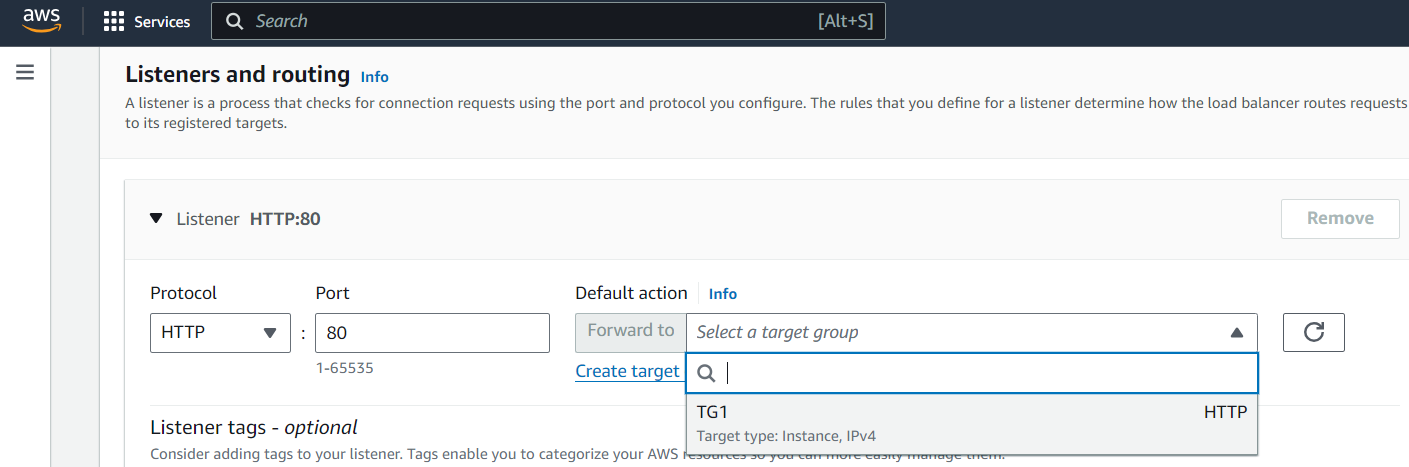


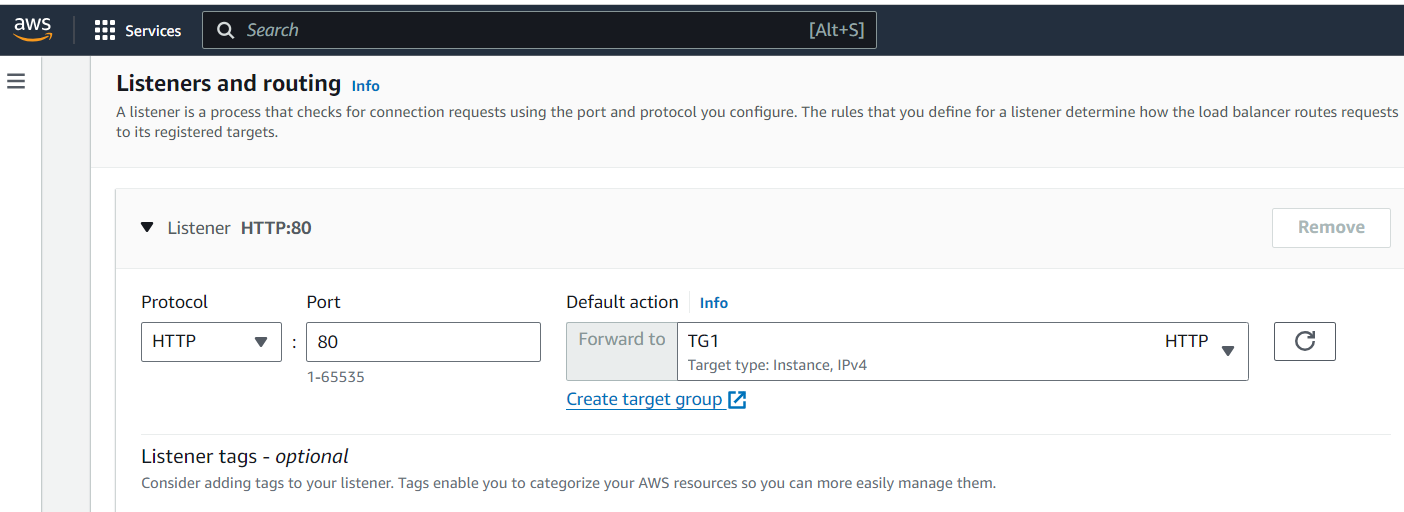


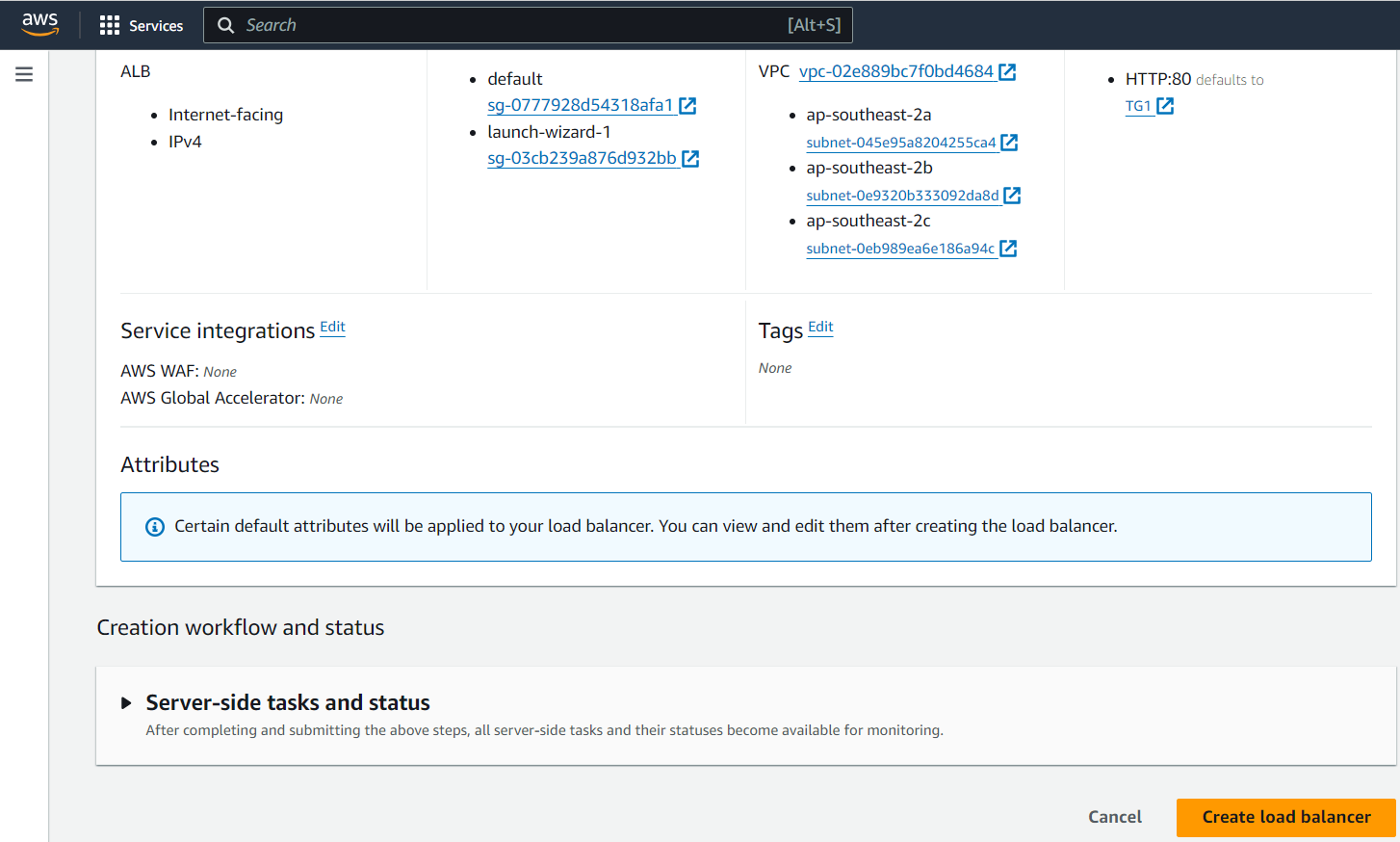


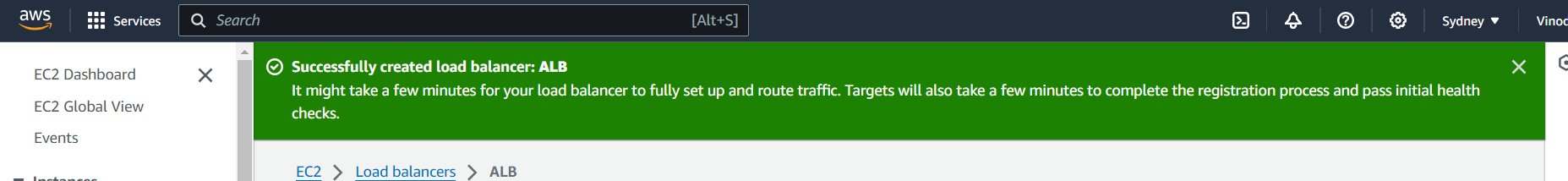


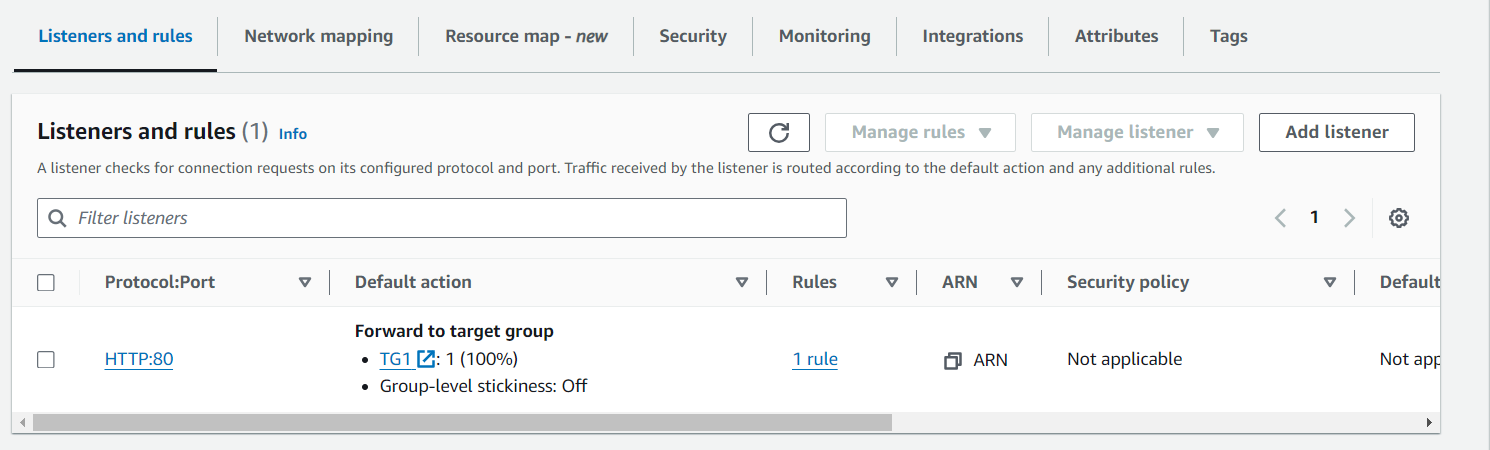


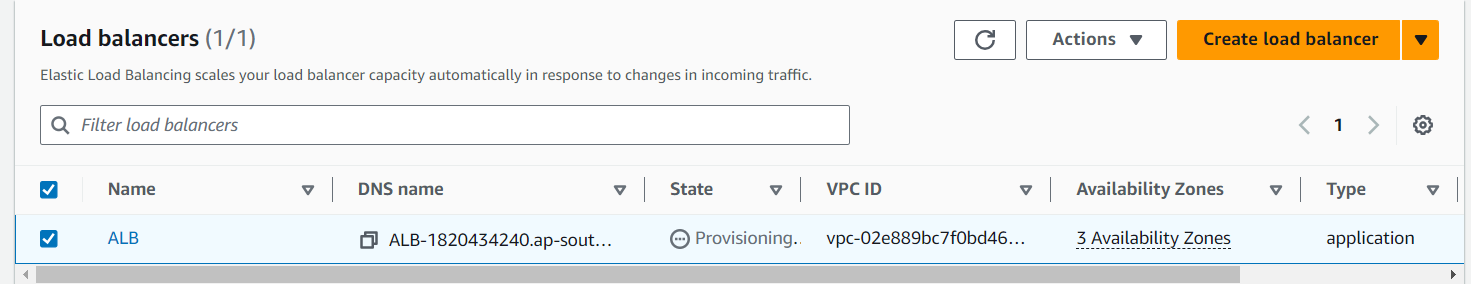


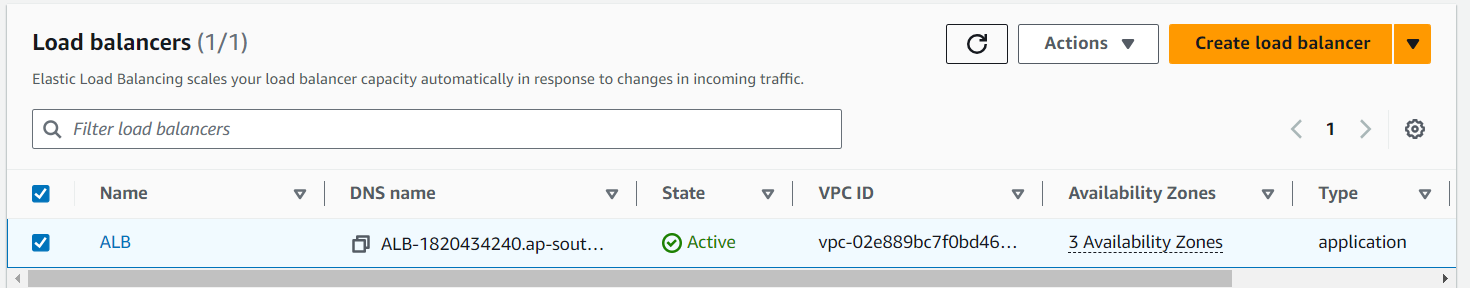




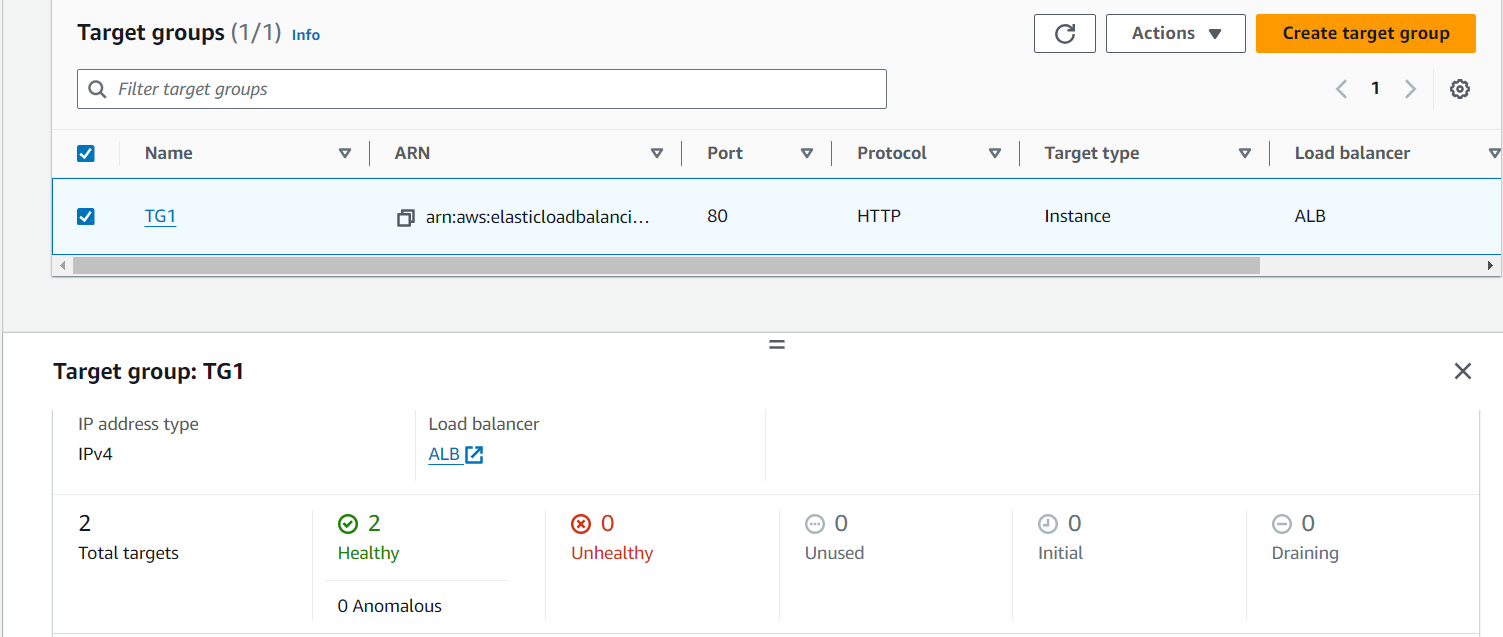




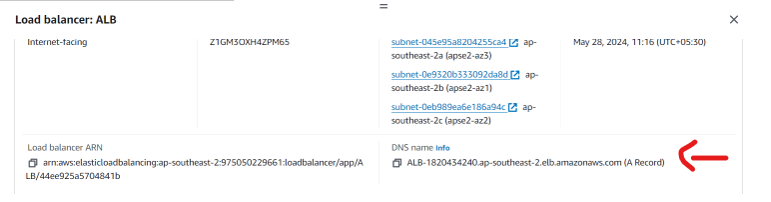




**Validating the Target group – Health check status**



**Load balancer - Common DNS**



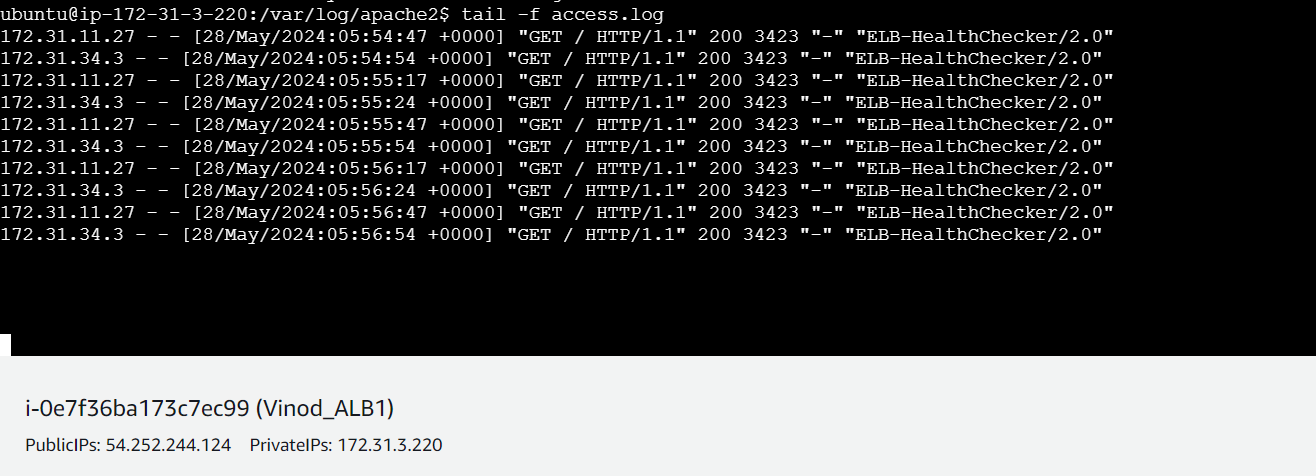
Now we can access both instance (application) by using common load balance link (DNS)

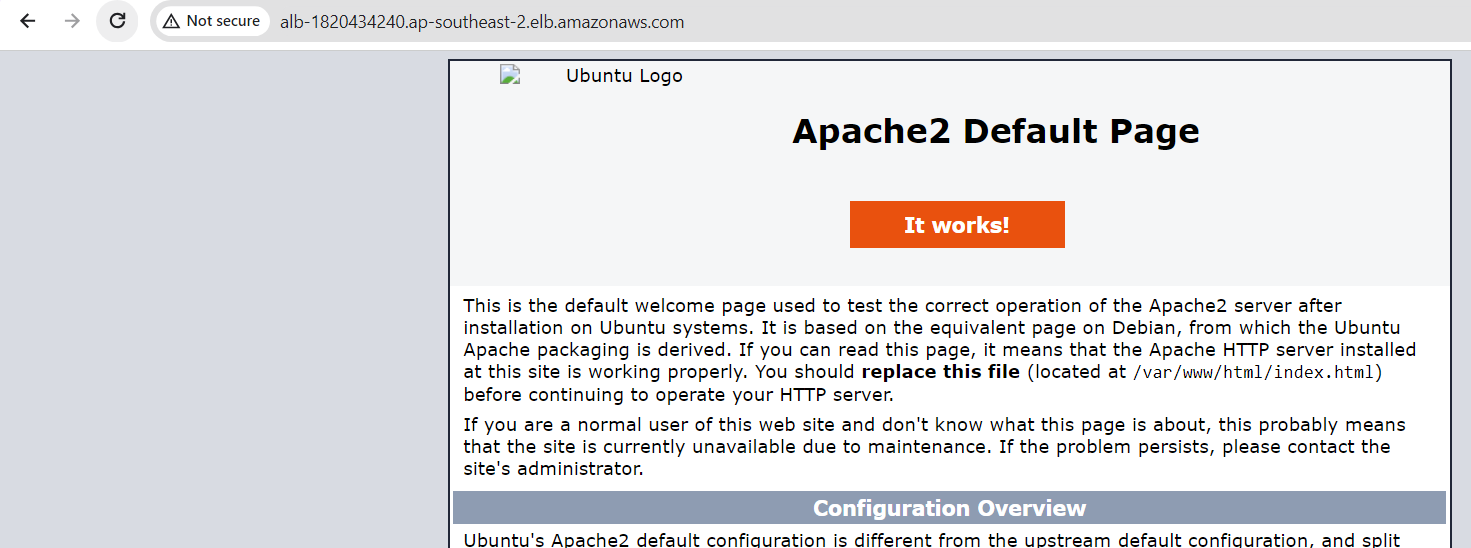
<http://alb-1820434240.ap-southeast-2.elb.amazonaws.com/>

Where the application gets loaded equally between two instances on each refresh or access

**Instance 1 ( Apache) :**

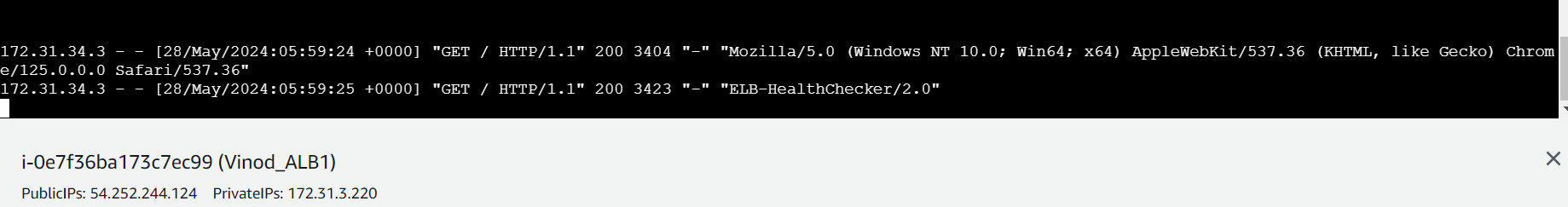
Before accessing the ALB DNS link





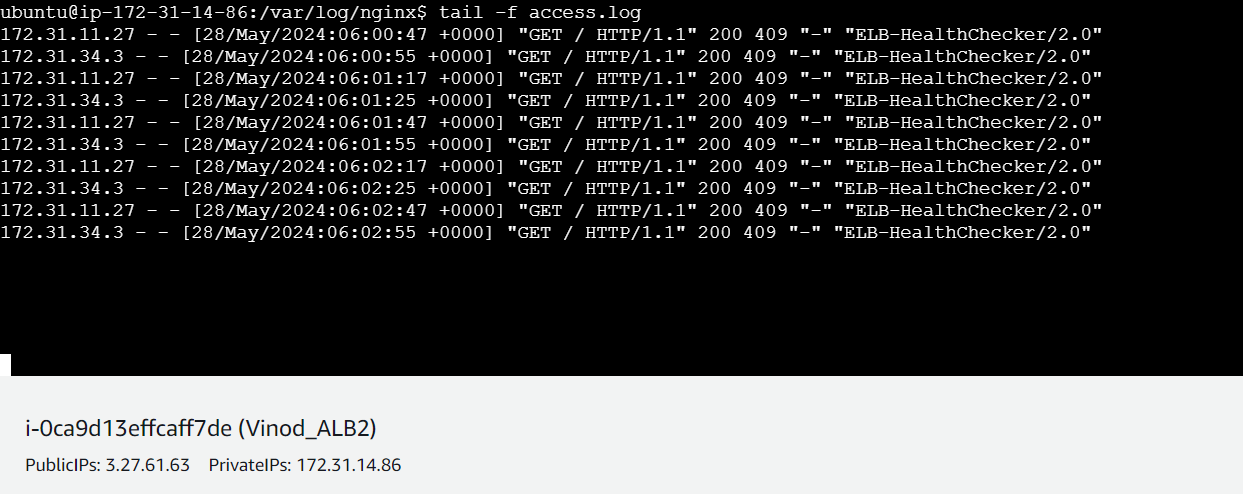
**Access entries has been captured in the Apache log**

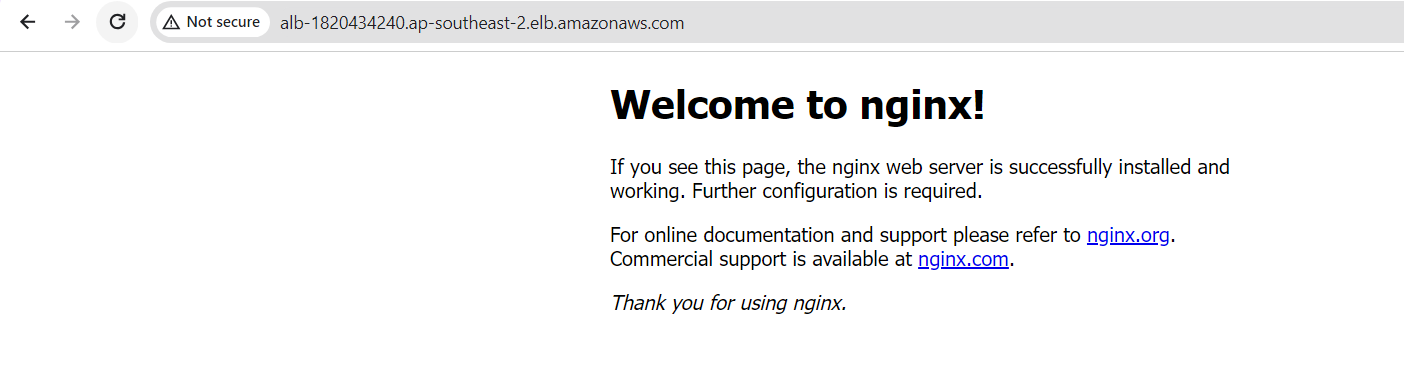
After accessing the ALB DNS link



**Instance 2 ( Nginx ) :**

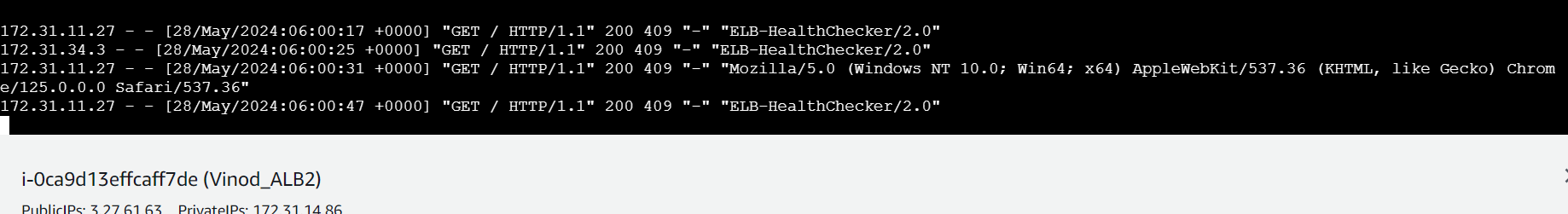
Before accessing the ALB DNS link





**Access entries has been captured the Nginx logs:**

After access



Hence we have successfully launched the EC2 Instances and the output traffic has been redirected via ALB DNS ” <http://alb-1820434240.ap-southeast-2.elb.amazonaws.com/>”