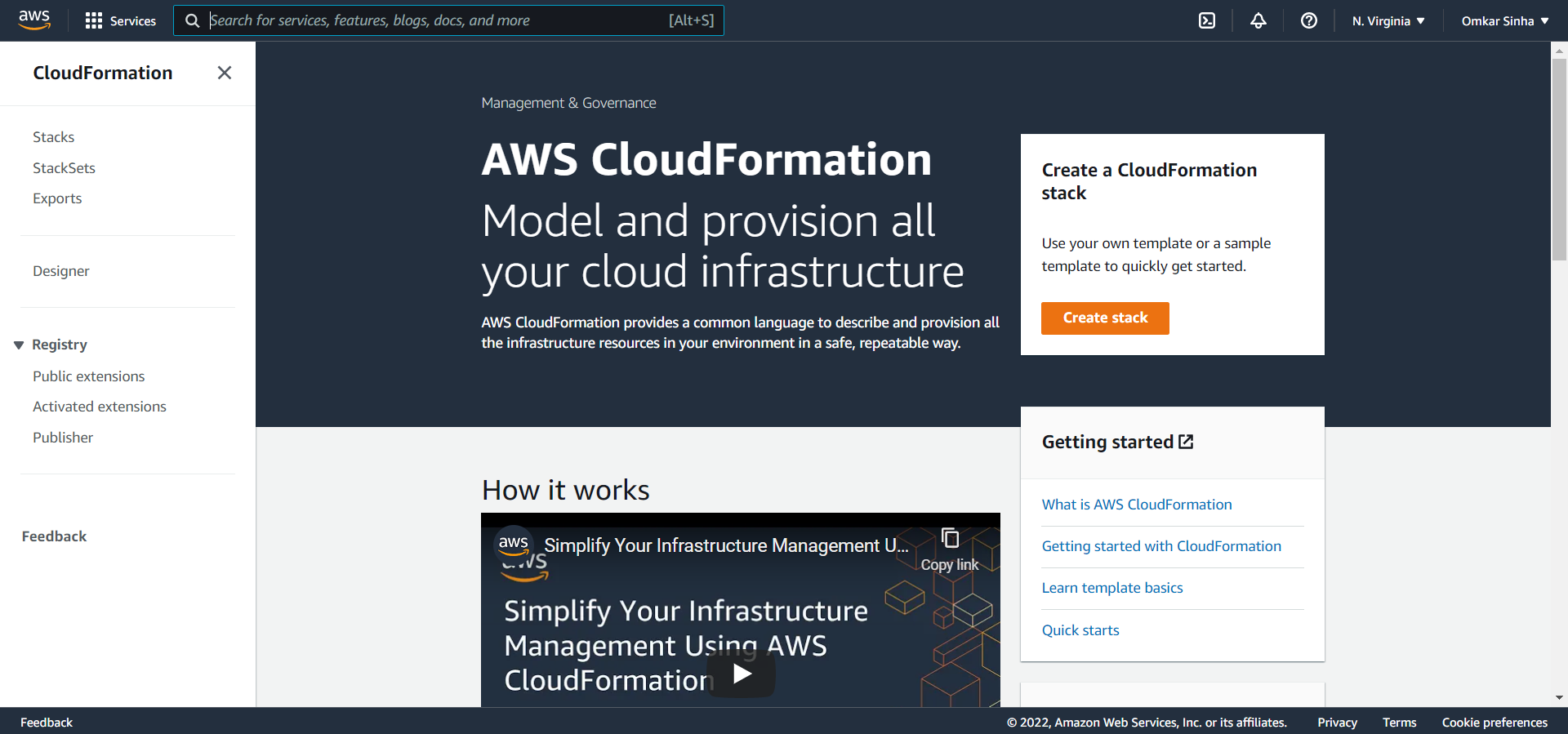
Lab – 3

1) First we visit AWS CloudFormation service:

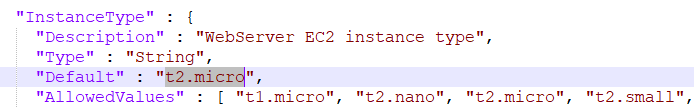


2) We create Stack template for the EC2 instances (named *Template\_For\_EC2\_instance.json*)

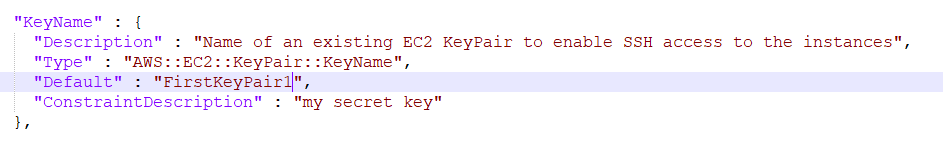
The sample template that we modified is: [*https://s3.amazonaws.com/cloudformation-templates-us-east-1/EC2InstanceWithSecurityGroupSample.template*](https://s3.amazonaws.com/cloudformation-templates-us-east-1/EC2InstanceWithSecurityGroupSample.template)

(the template used is submitted along with report)

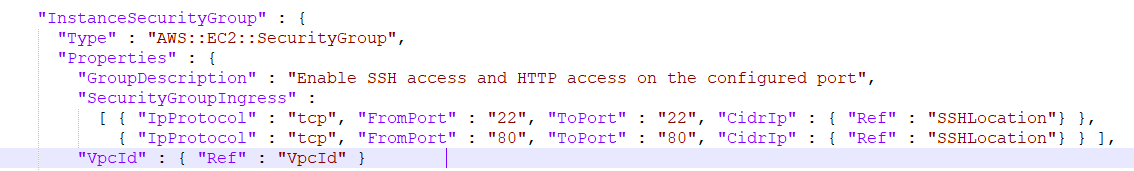
i) We change to t2.micro instance since its free:



ii) We change on the template to put our key:



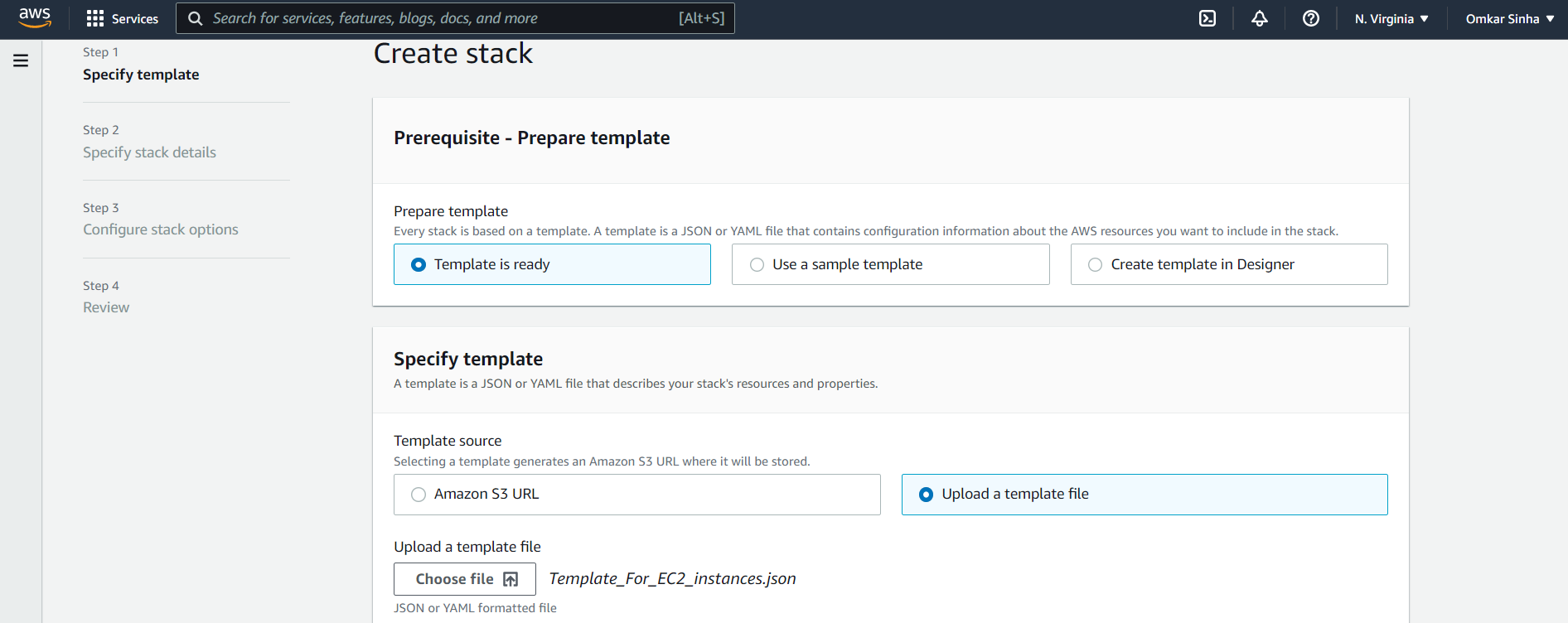
iii) We change security group to include port 80 for http:



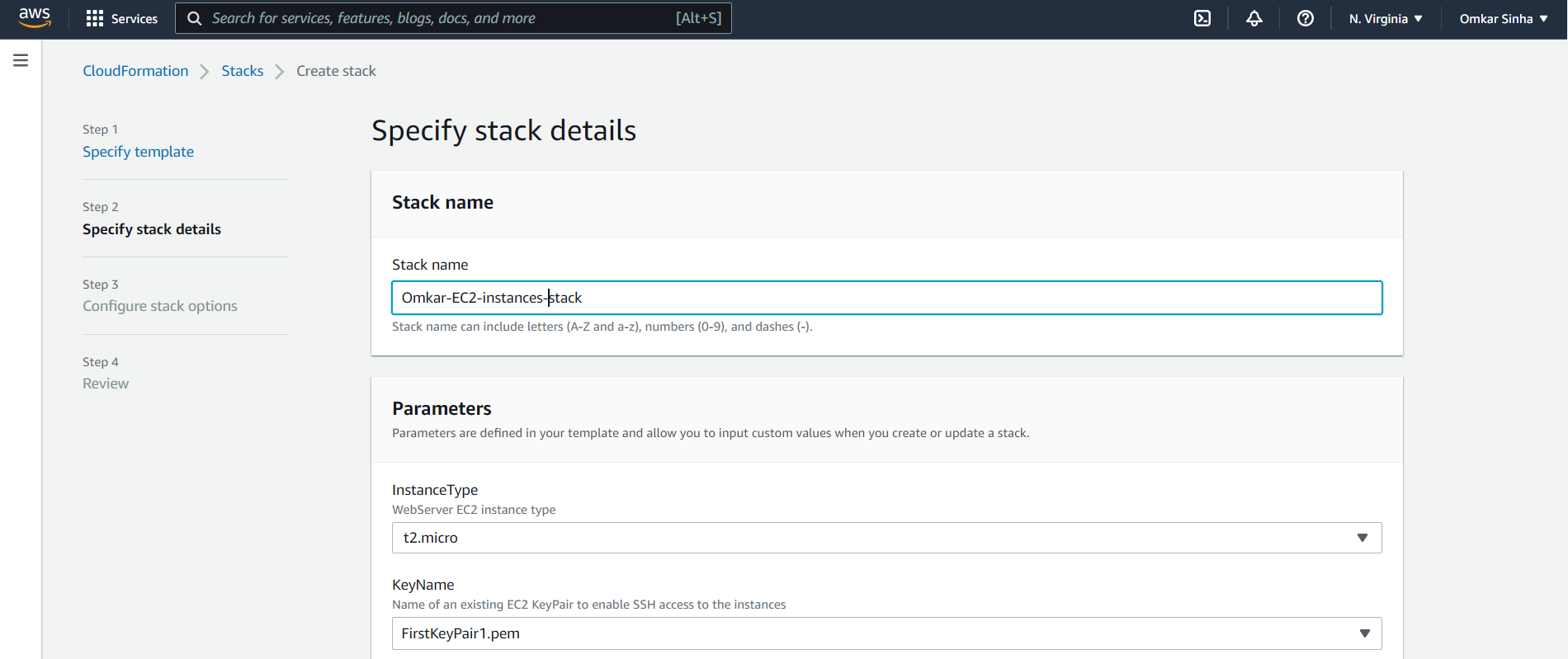
iv) We keep the CIDR group unchanged

3) Now we create the stack

i) First, we specify the template. Here we upload the template from our local system.



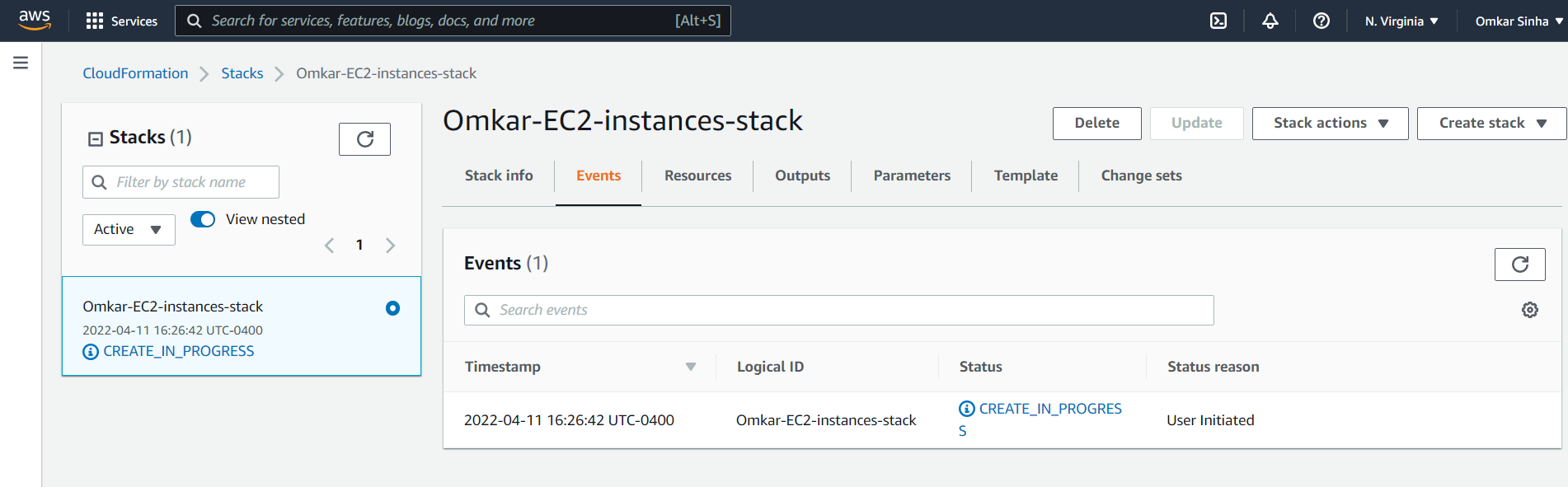
ii) Next, we specify stack details. Here we only given a stack name and keep rest unchanged.



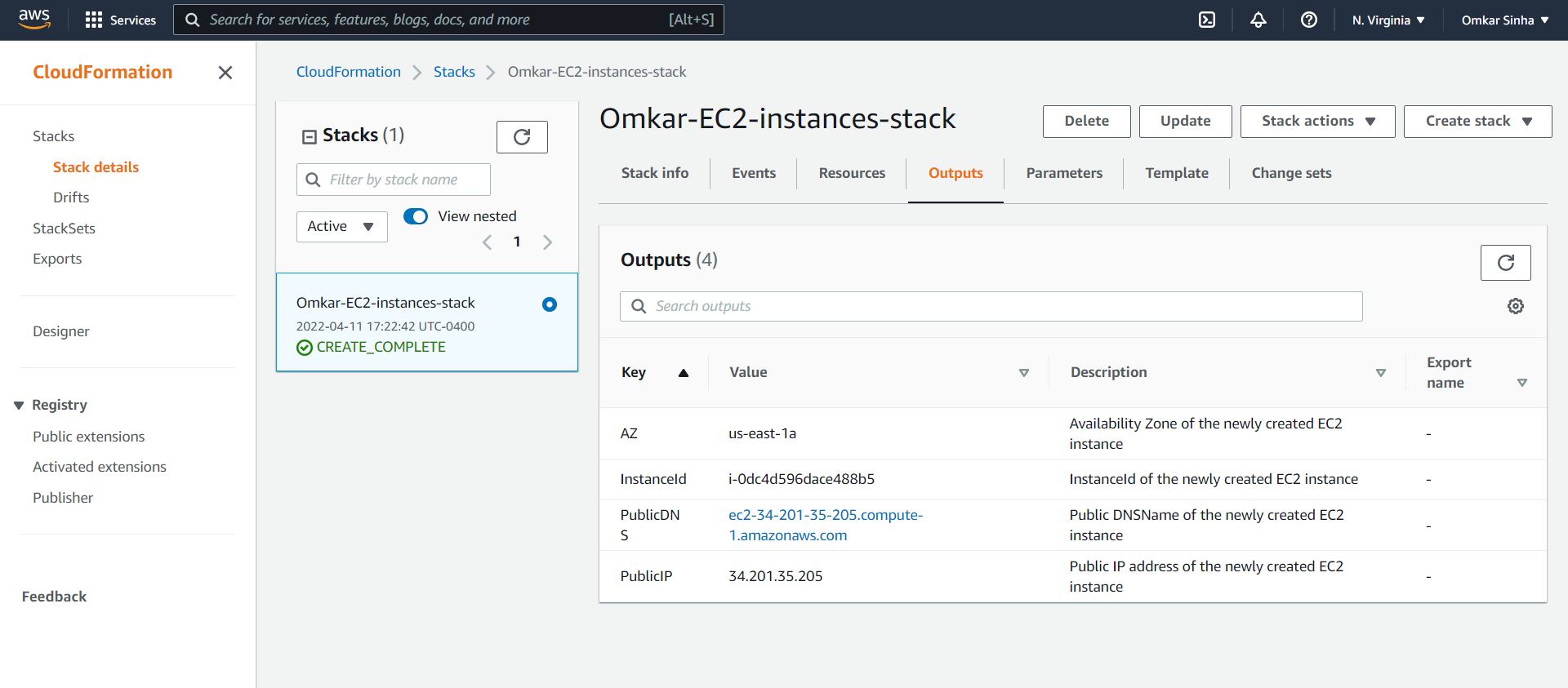
iii) We don’t change anything in “configure stack details” and then review it.

3)

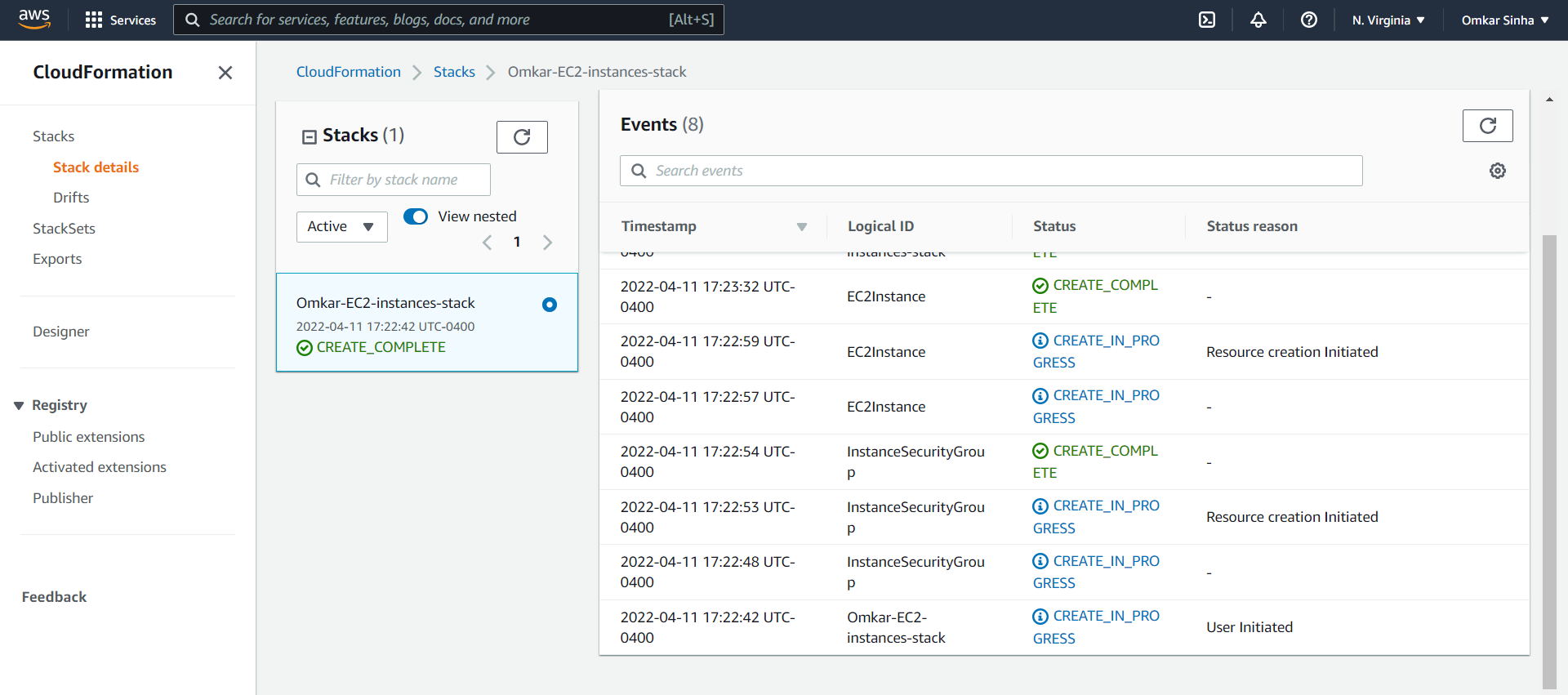
i) We choose create stack. It will “create in progress” for the duration of the create.



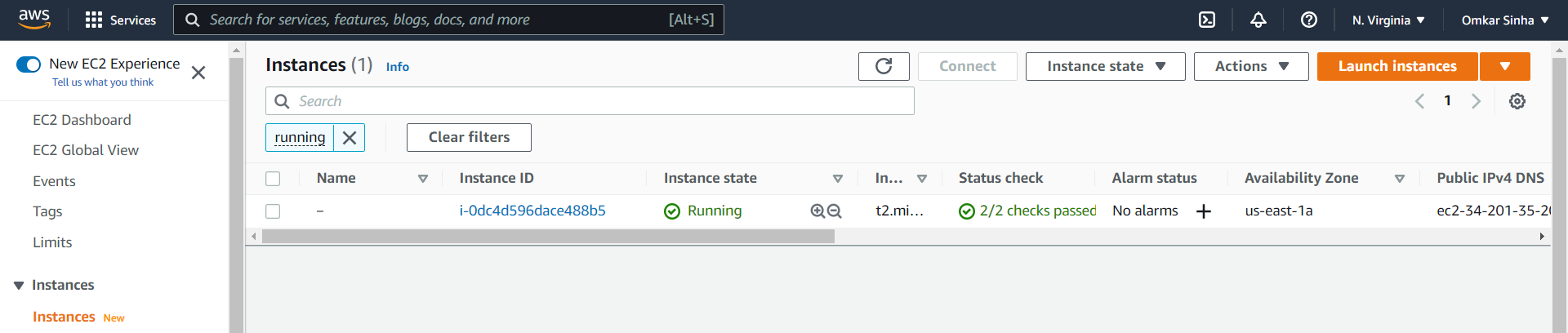
ii) The create is complete. The output of the newly created stack is shown below:



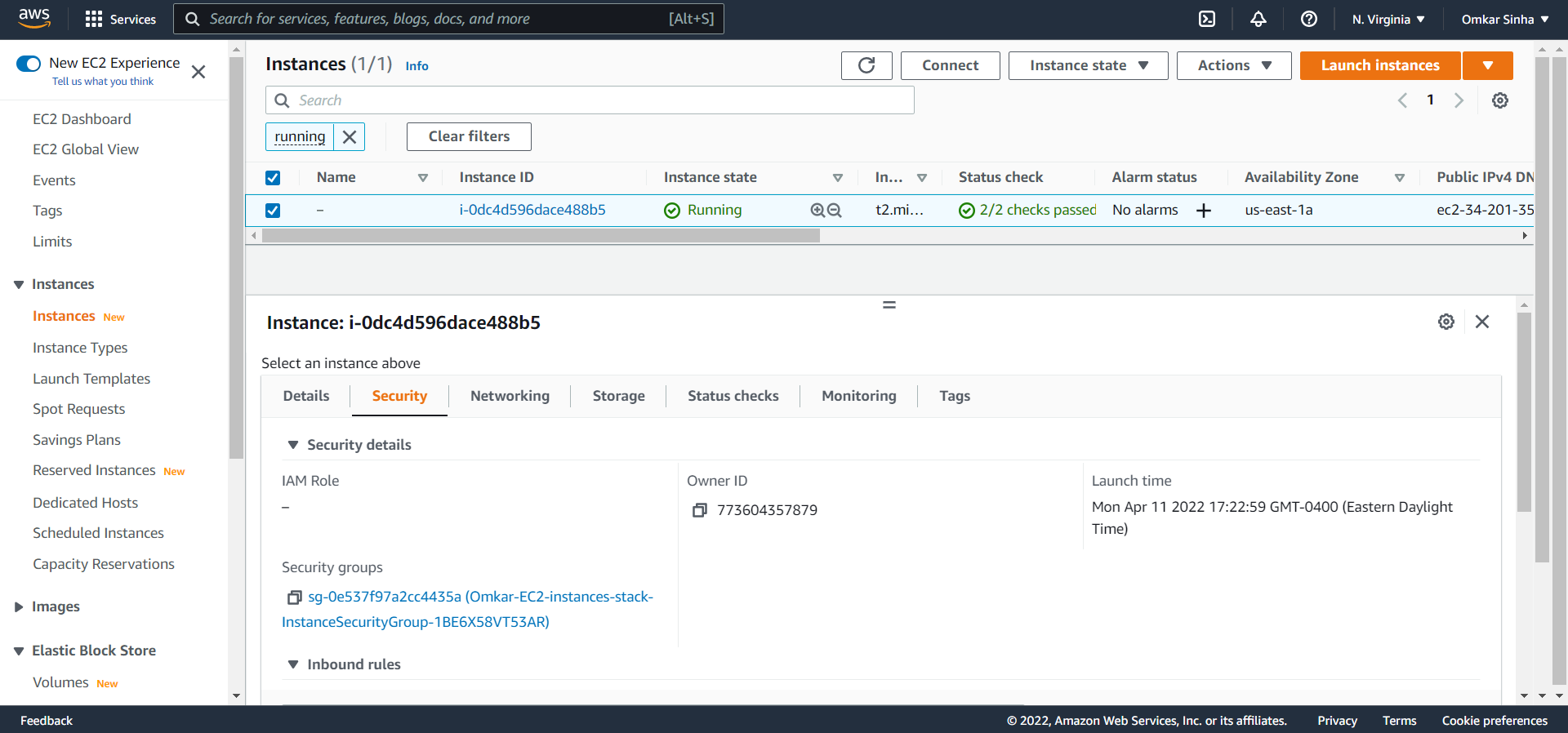
iii) We can also view the events to create the stack

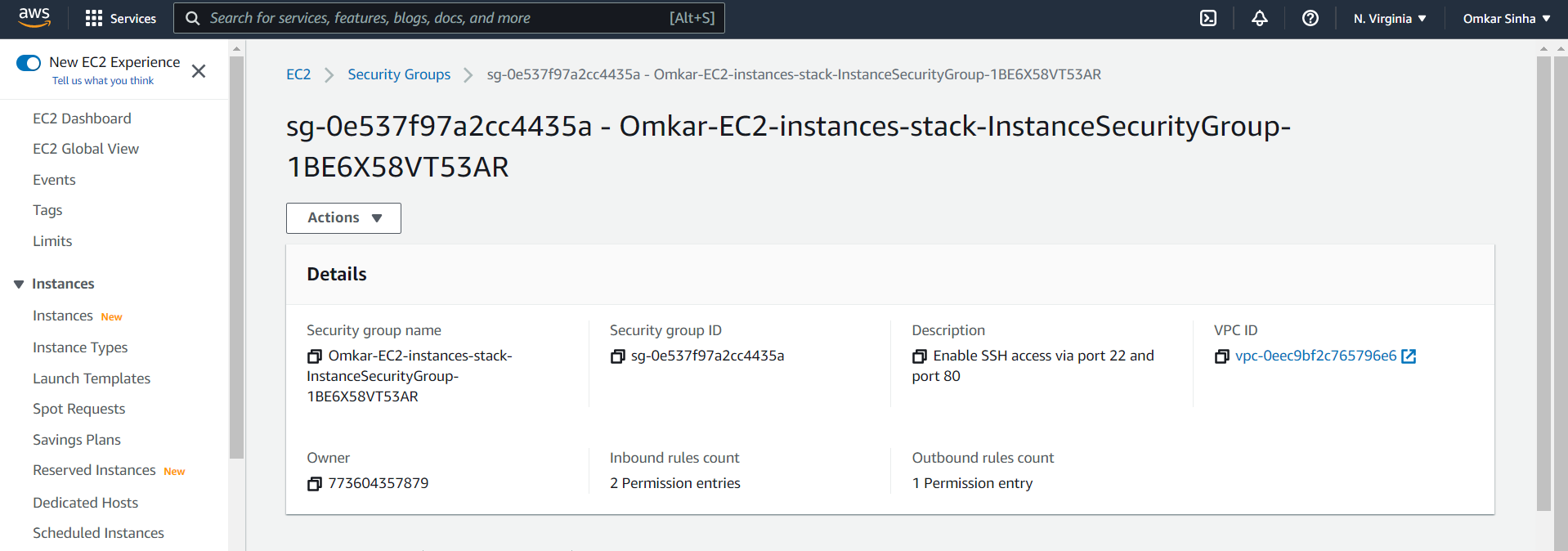


iv) We notice that 1 EC2 instance is deployed as a result of the stack creation.

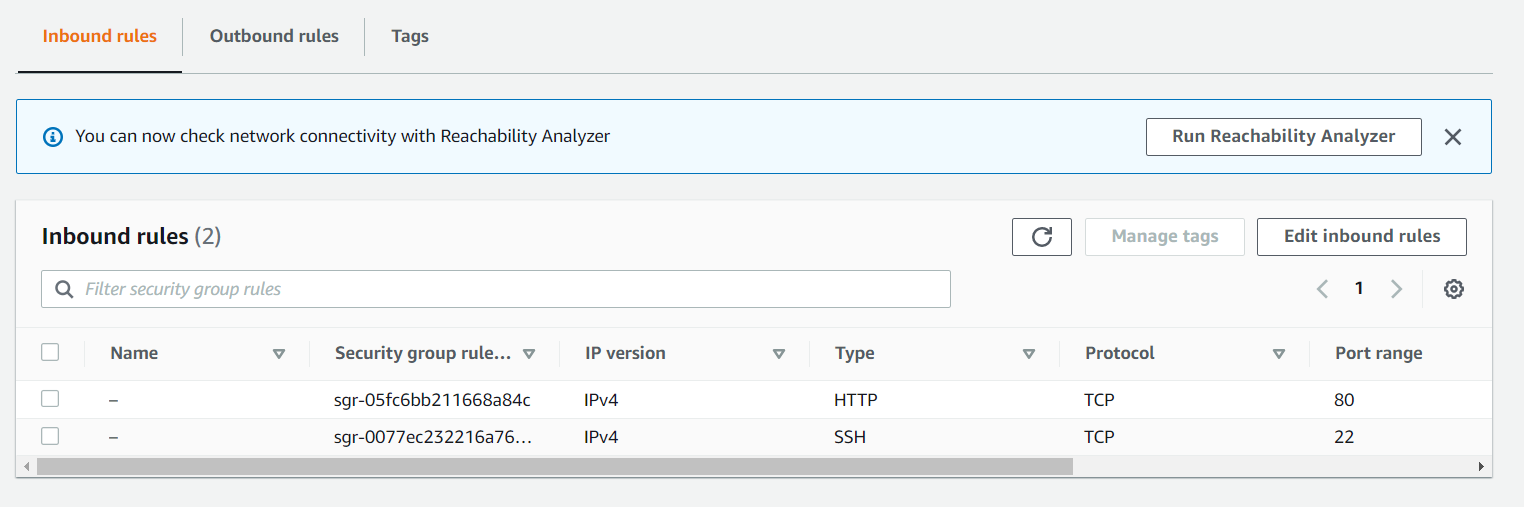


v) We also check the security groups





vi) And we verify the network interface for the deployed instances



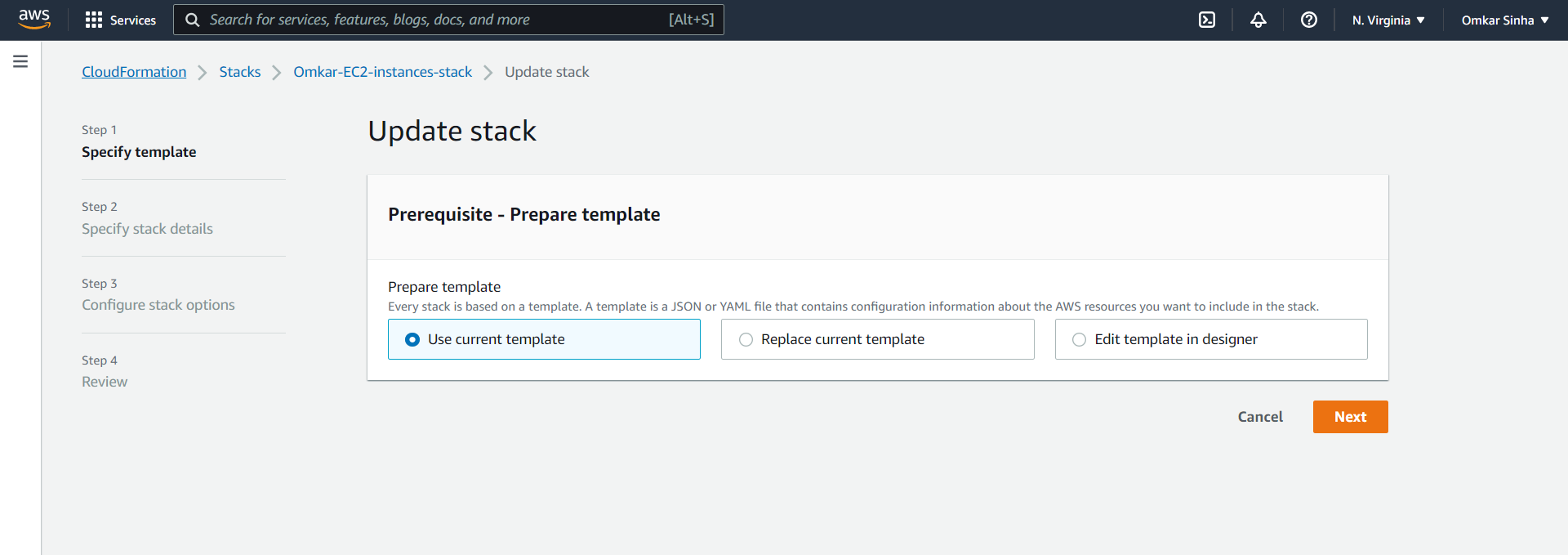
4) Now we update the existing stack to create a Load Balancer. We also deploy currently 2 instances.

(Note that we choose to keep instances at 2 right now instead of 4. We will later implement scaling and change to 4 to check version constrolling)

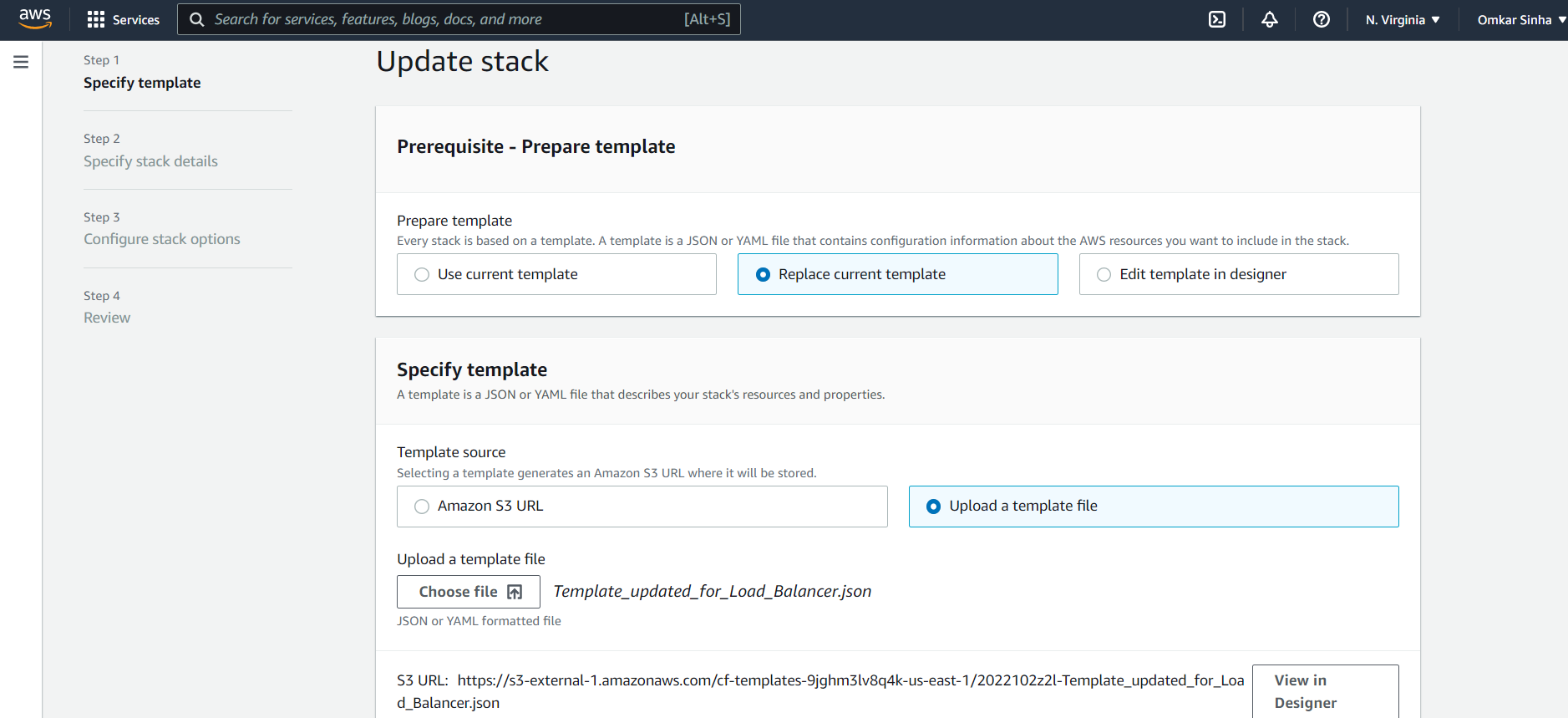
The sample template we used as reference is : [*https://s3.amazonaws.com/cloudformation-templates-us-east-1/AutoScalingKeepAtNSample.template*](https://s3.amazonaws.com/cloudformation-templates-us-east-1/AutoScalingKeepAtNSample.template)

(the template used is submitted along with report)

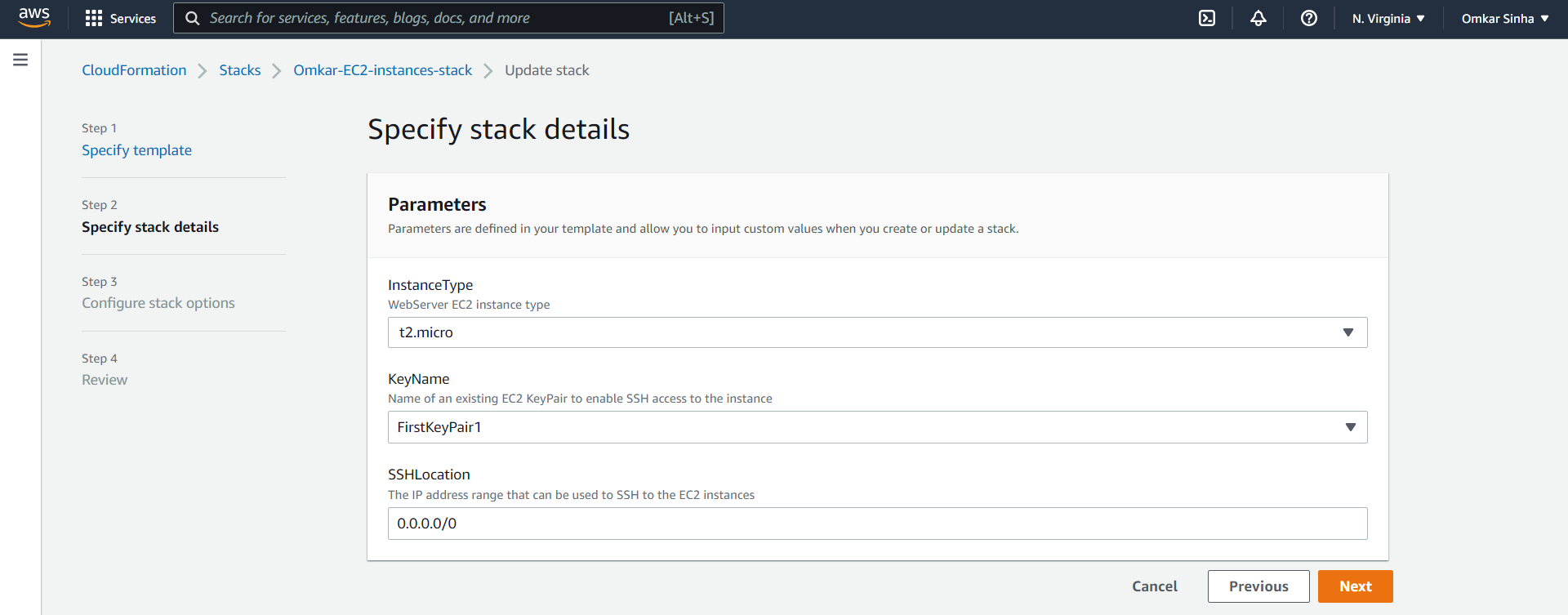
i) We choose the update stack option :



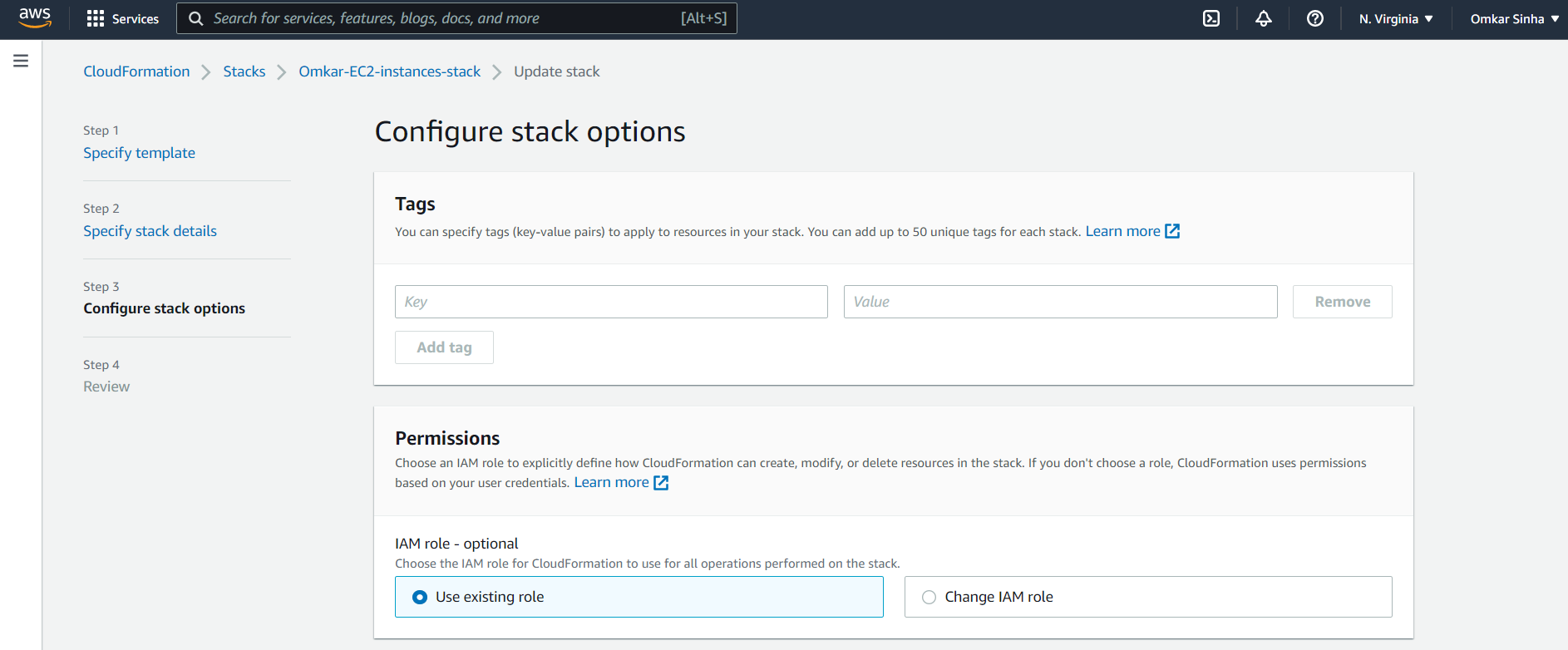
ii) Updating stack by choosing “Replace stack” and then “Upload new stack”, from above dialog box:



iii) We specify the stack details:

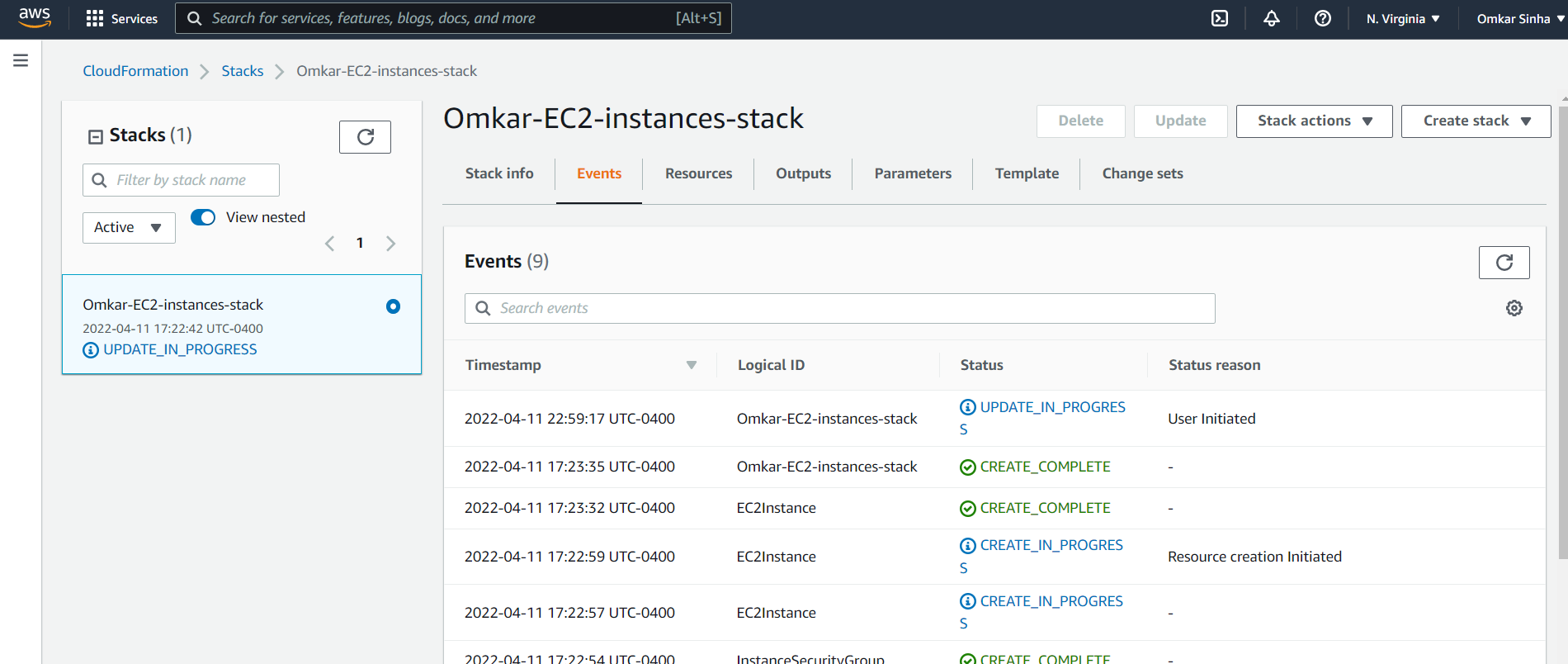


iii) We keep the configure fields unchanged:

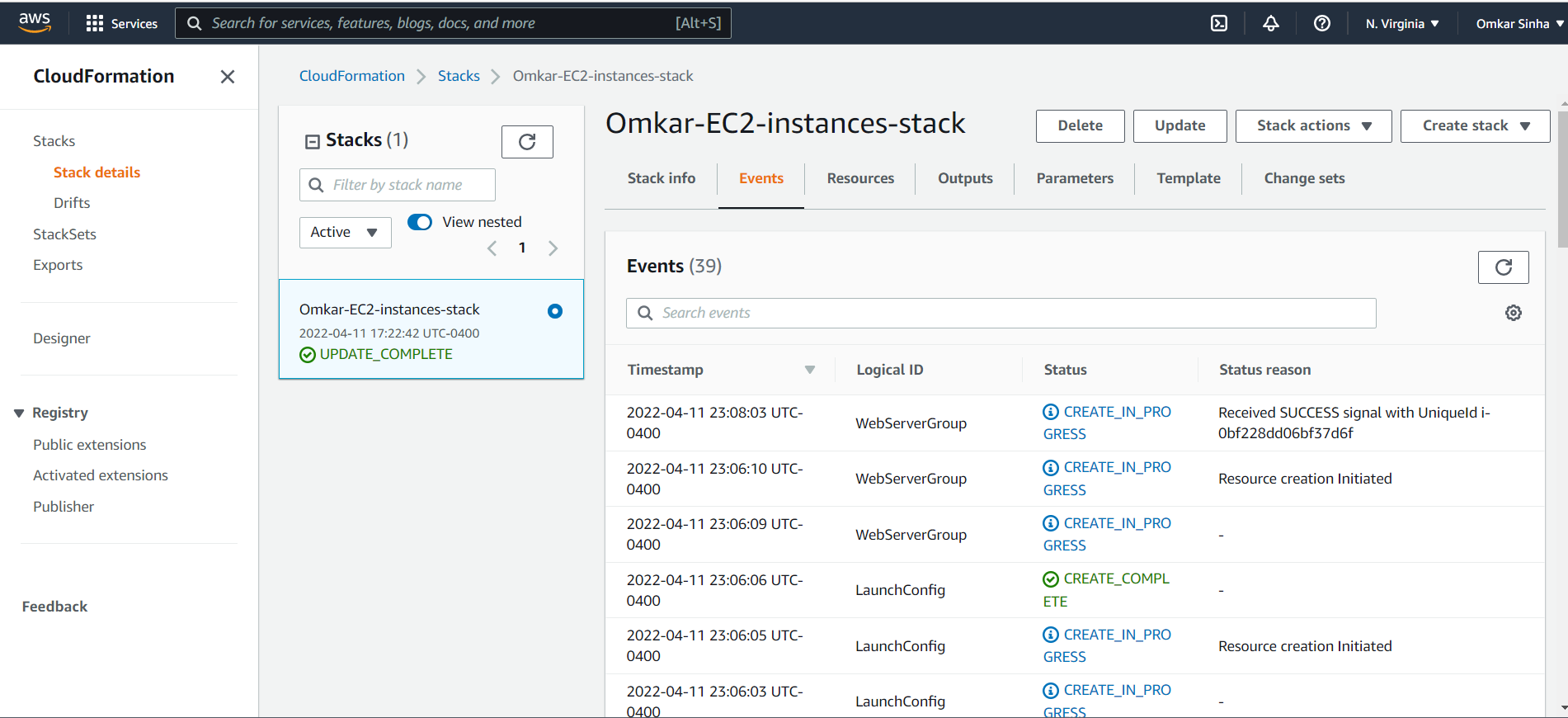


5) We select Update.

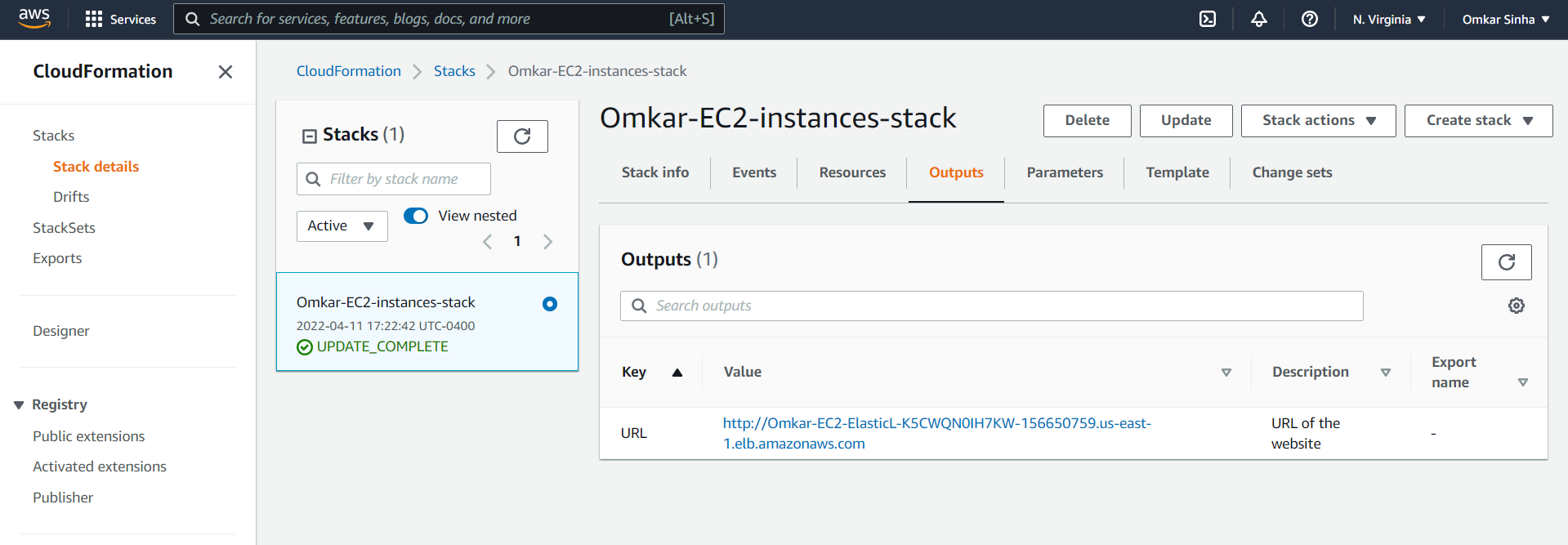
i) Update is in progress:



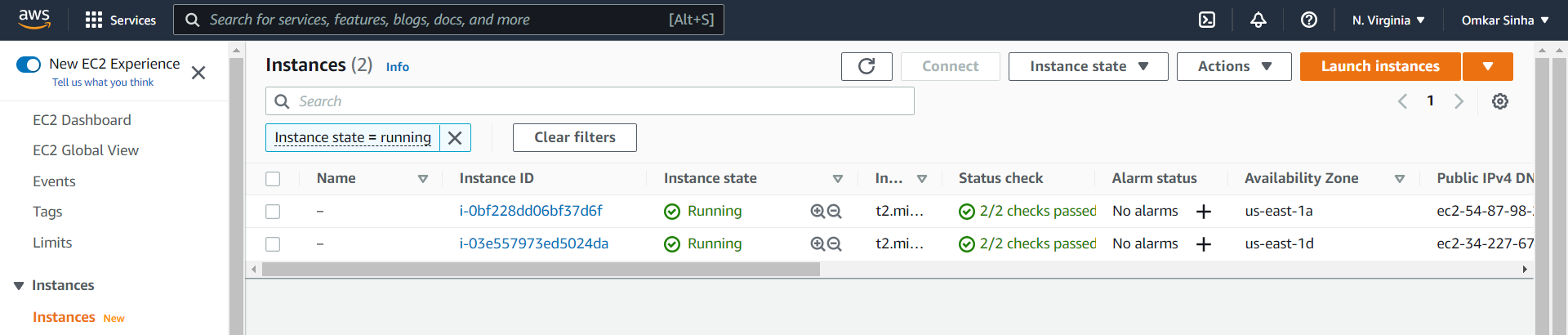
ii) Update successfully completes:

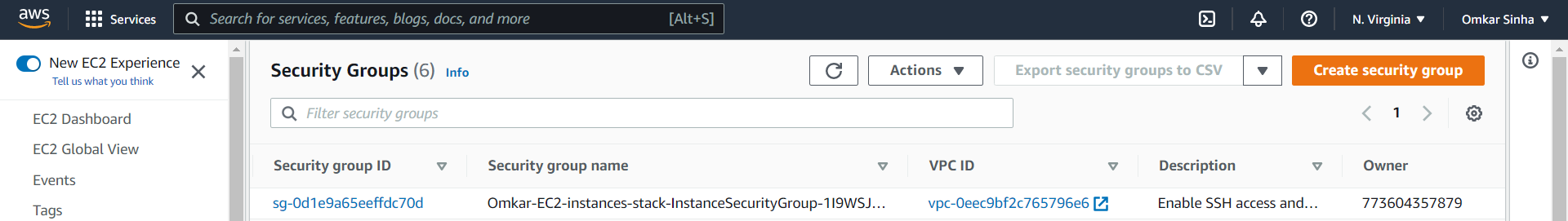


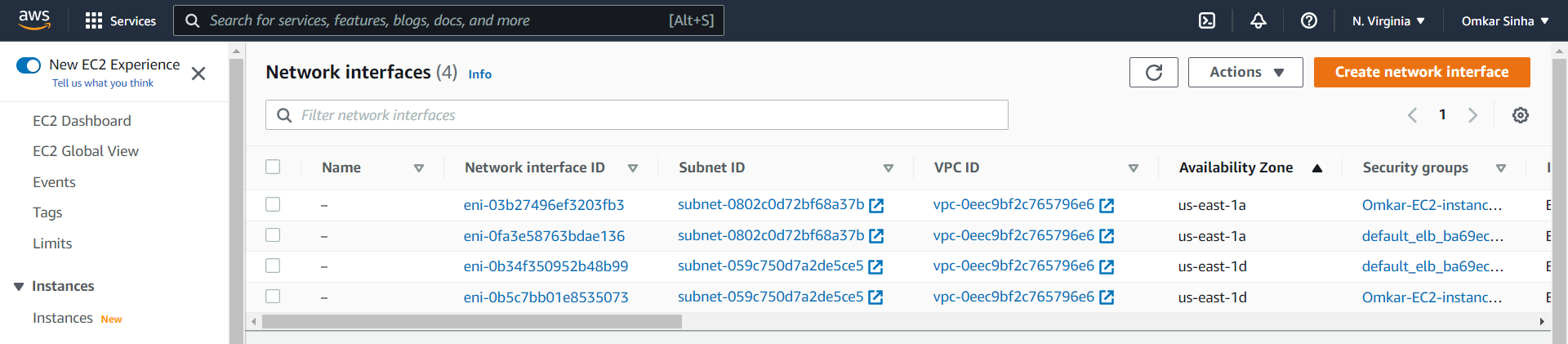
iii) Check output of updated stack



iv) Now verifying that ec2 instances, security groups and network interfaces are successfully created in that order as follows respectively:

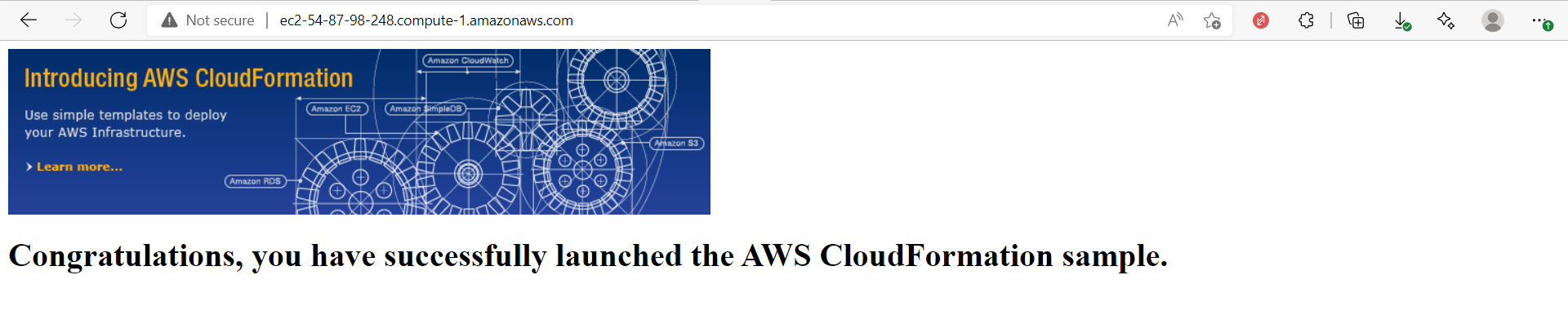




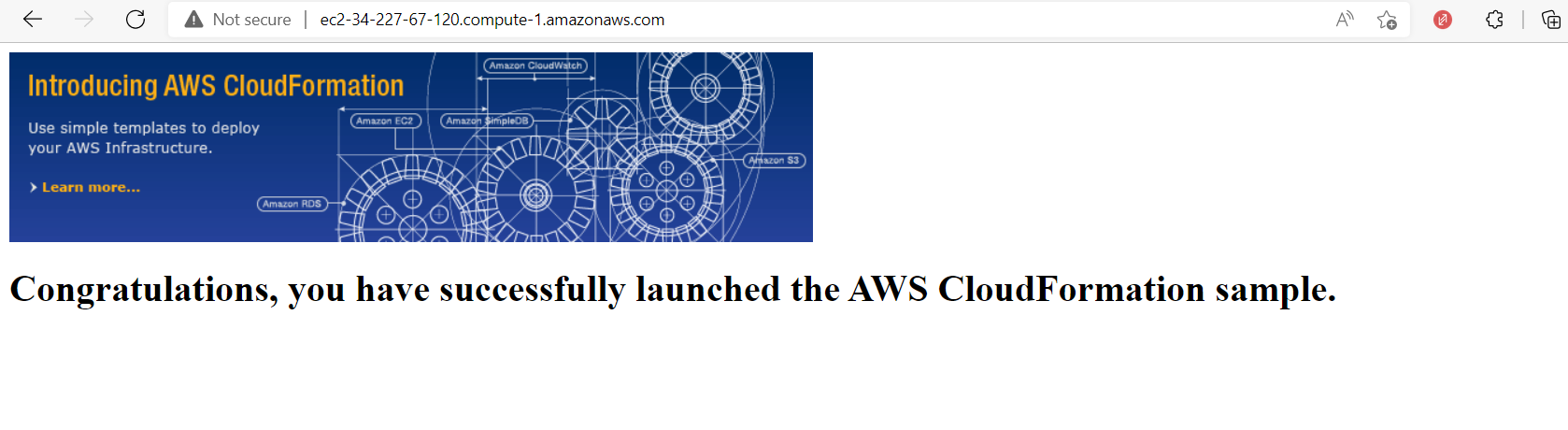


v) We open the public DNS link for each of the servers.

Server 1:

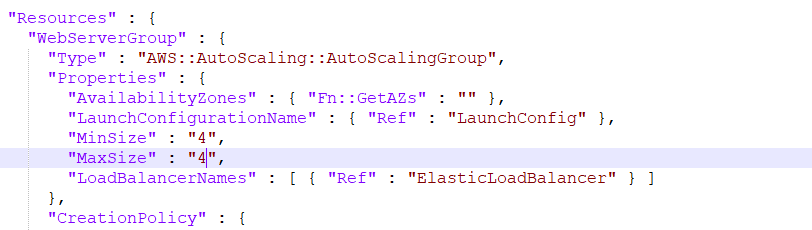


Server 2:

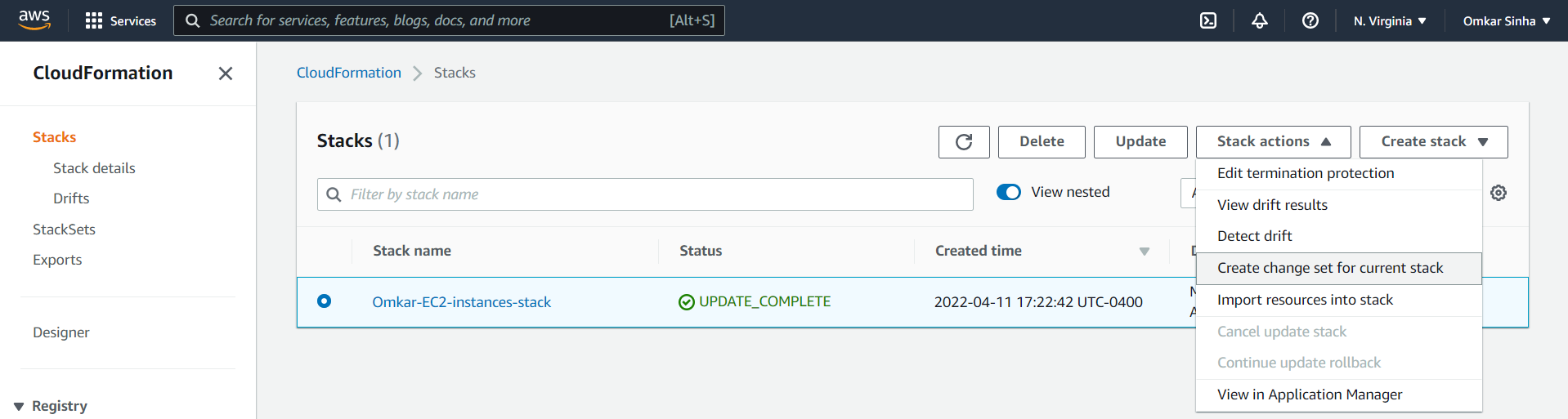


6) ADDITIONAL STEP:

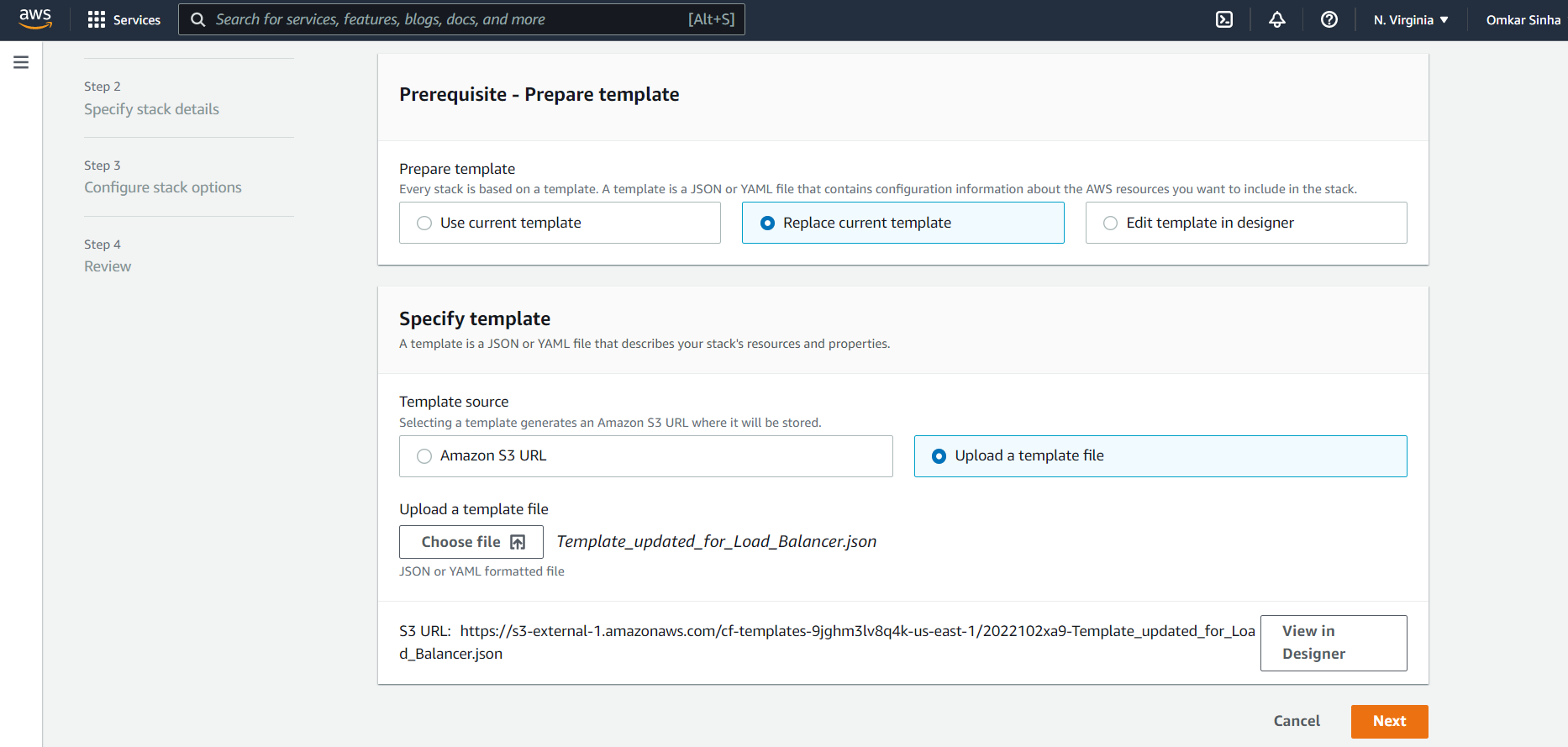
Now, we increase the total instances to 4. We create a CREATE CHANGE SET for autoscaling the instances to 4 to also check version controlling. We make following changes in the load balancer template.



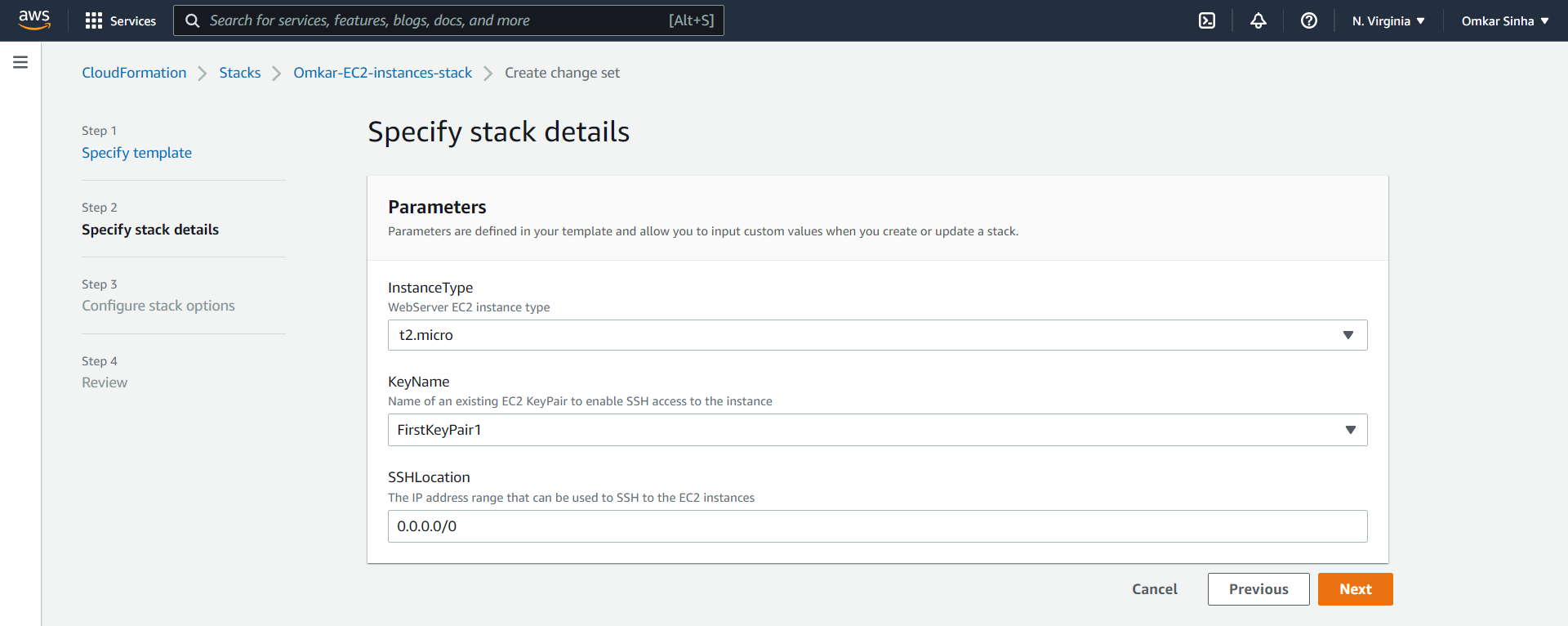
i) Now we choose create change set from stack actions menu:



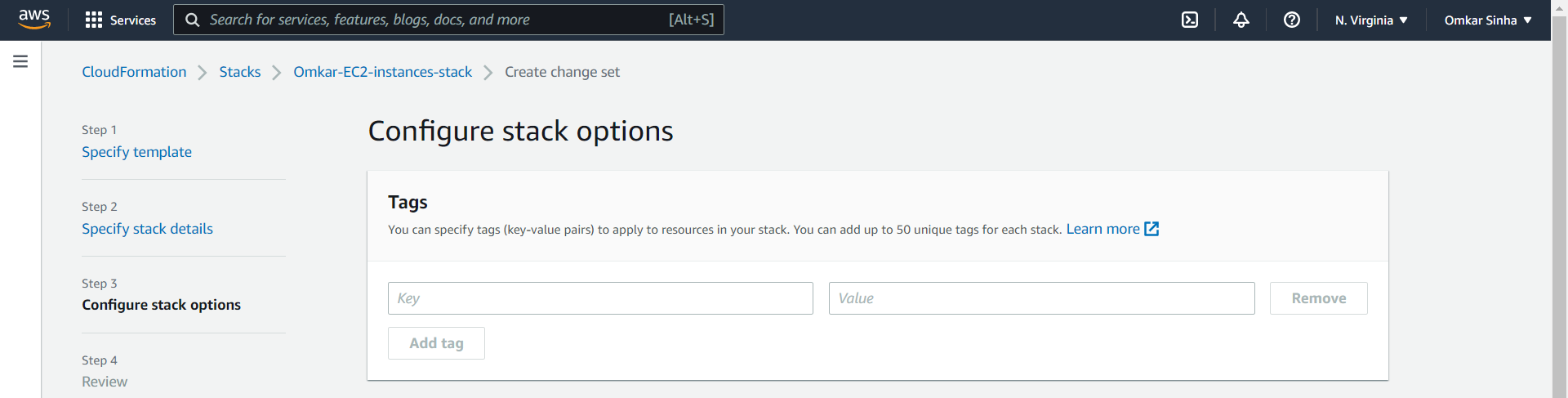
ii) We upload the update template.



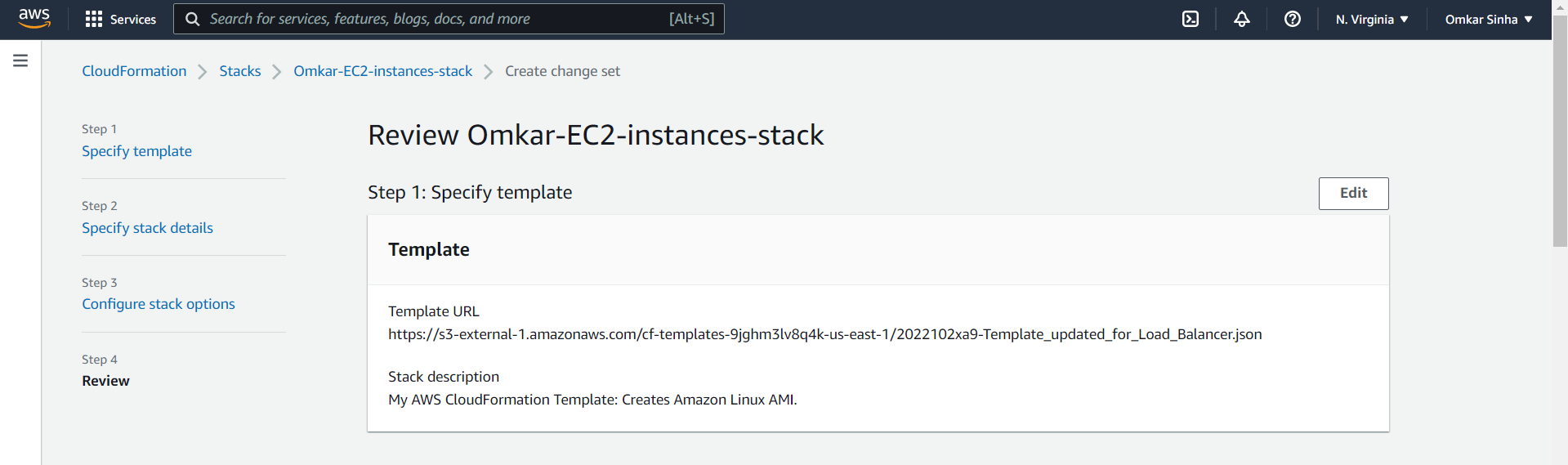
iii) We specify the stack details.



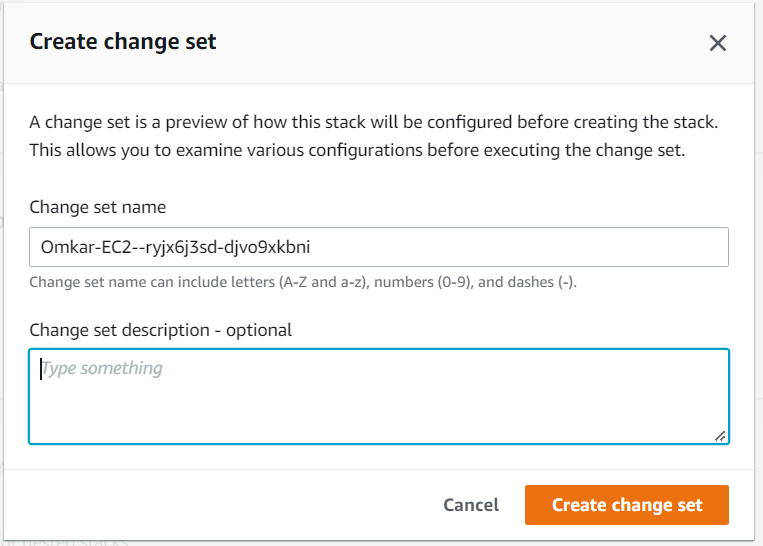
iv) We keep configuration unchanged:

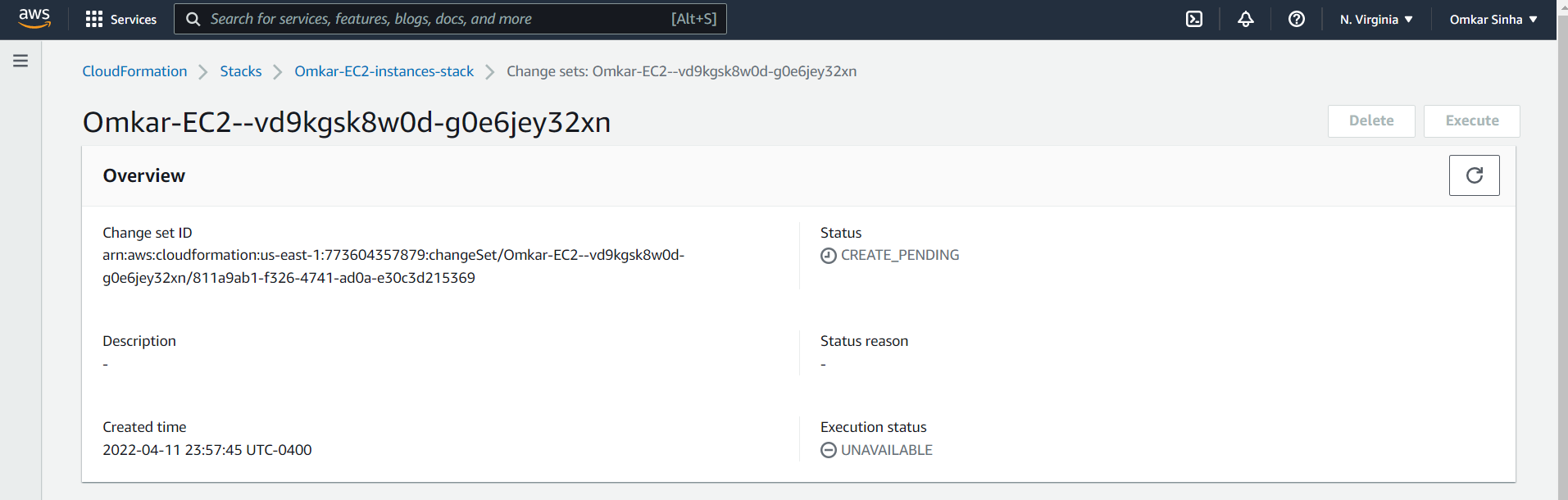


v) We review the changes:

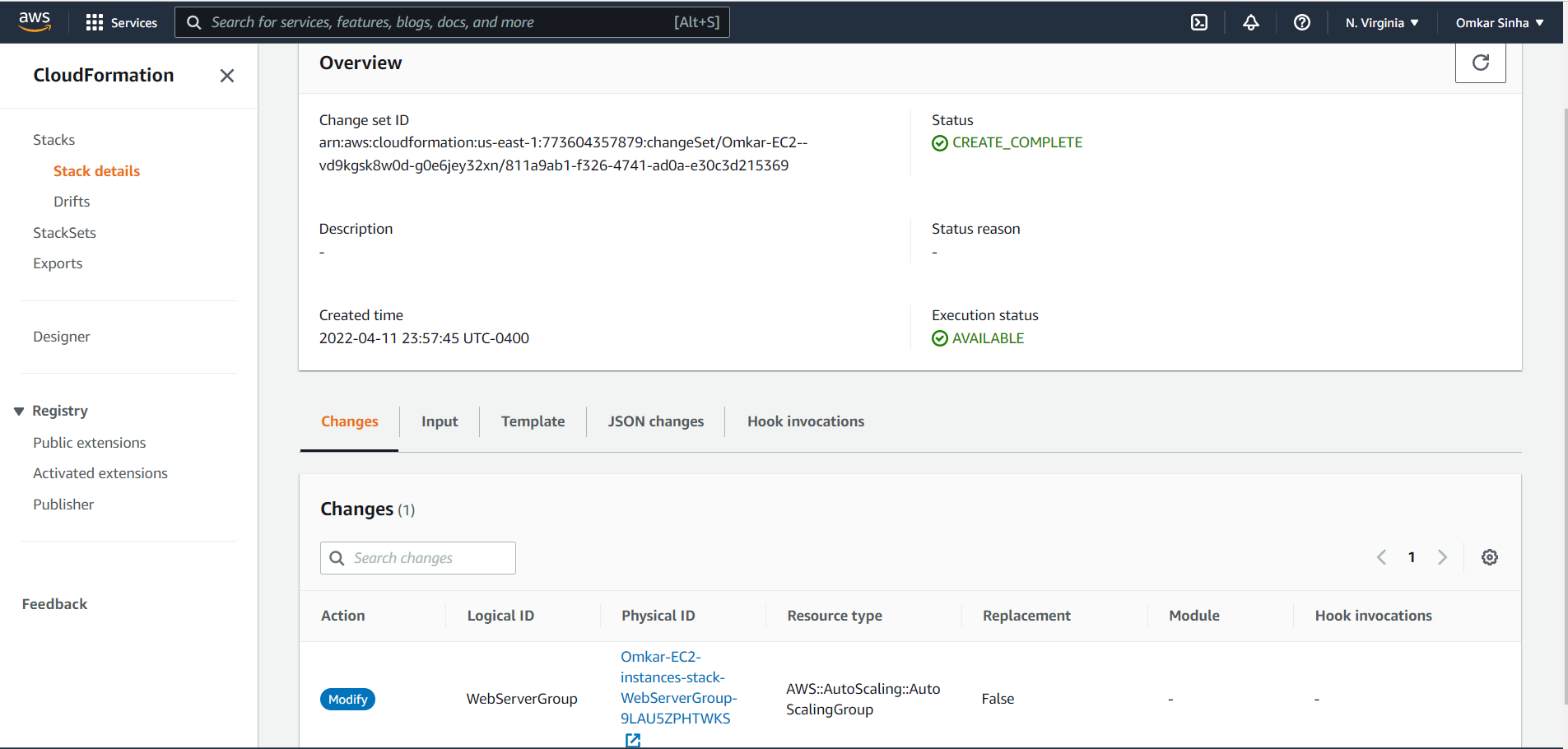


v) We create the change set:

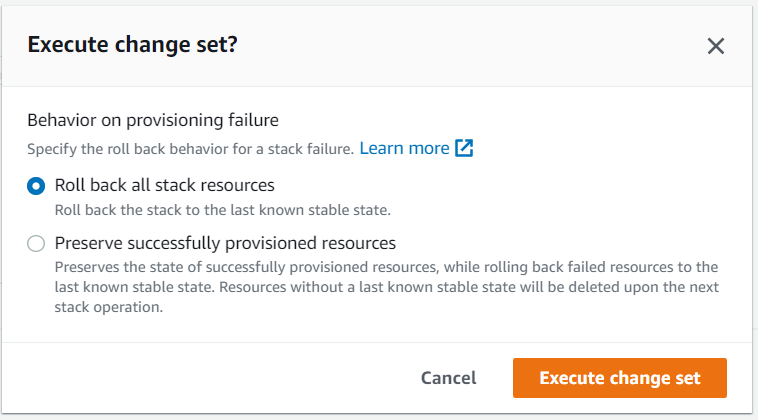




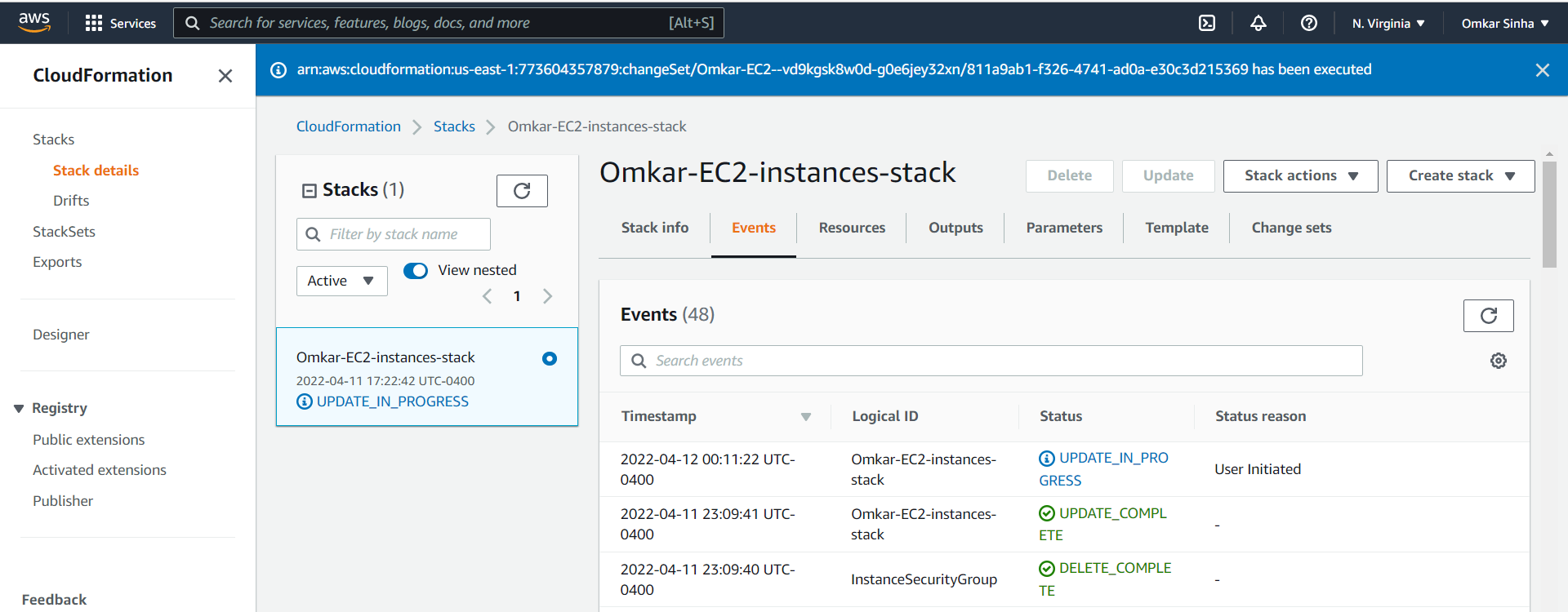
vi) Now, we check the update stack details and notice that WebServerGroup is modified rather than replaced.



vii) We execute the change set:



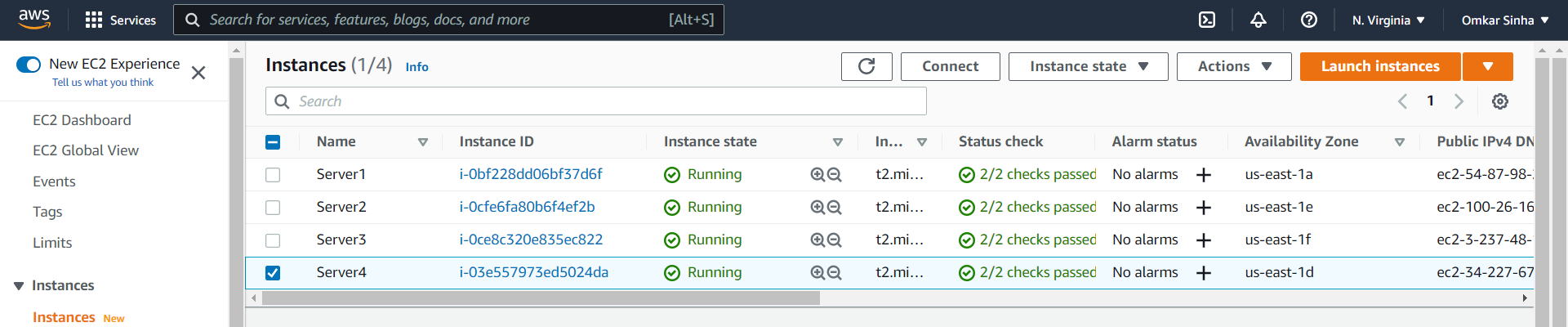
viii) Executing change set update in progress:

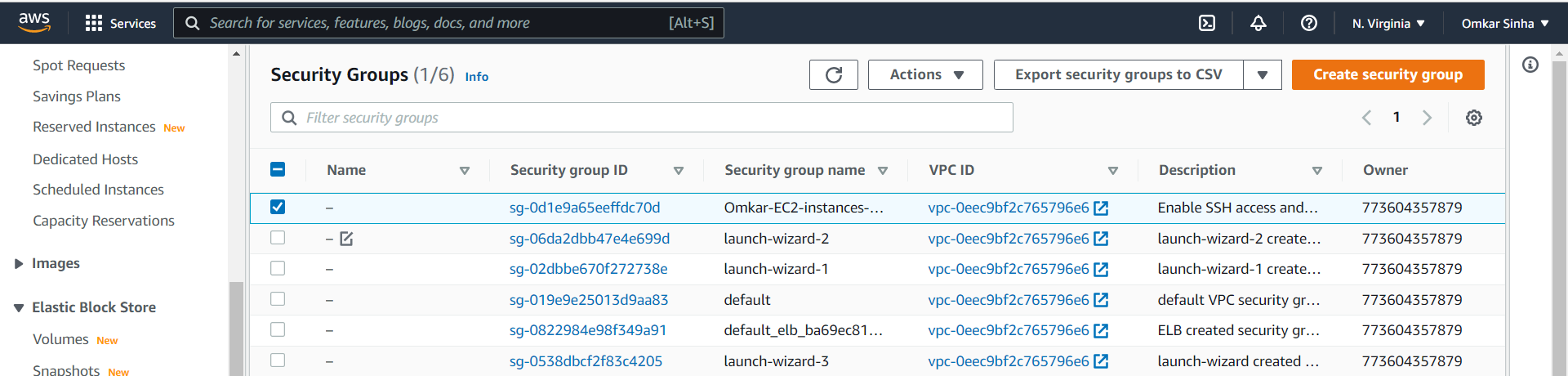


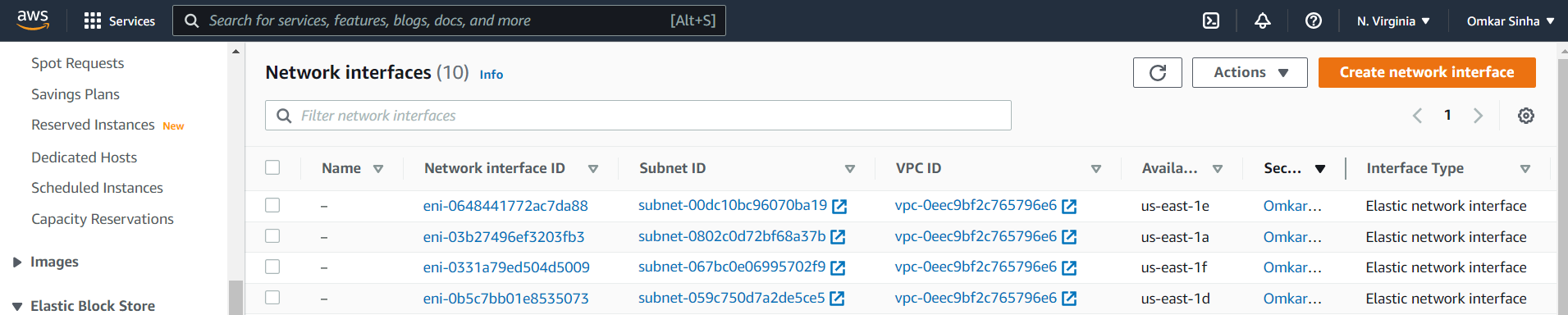
ix) We find the update successful:



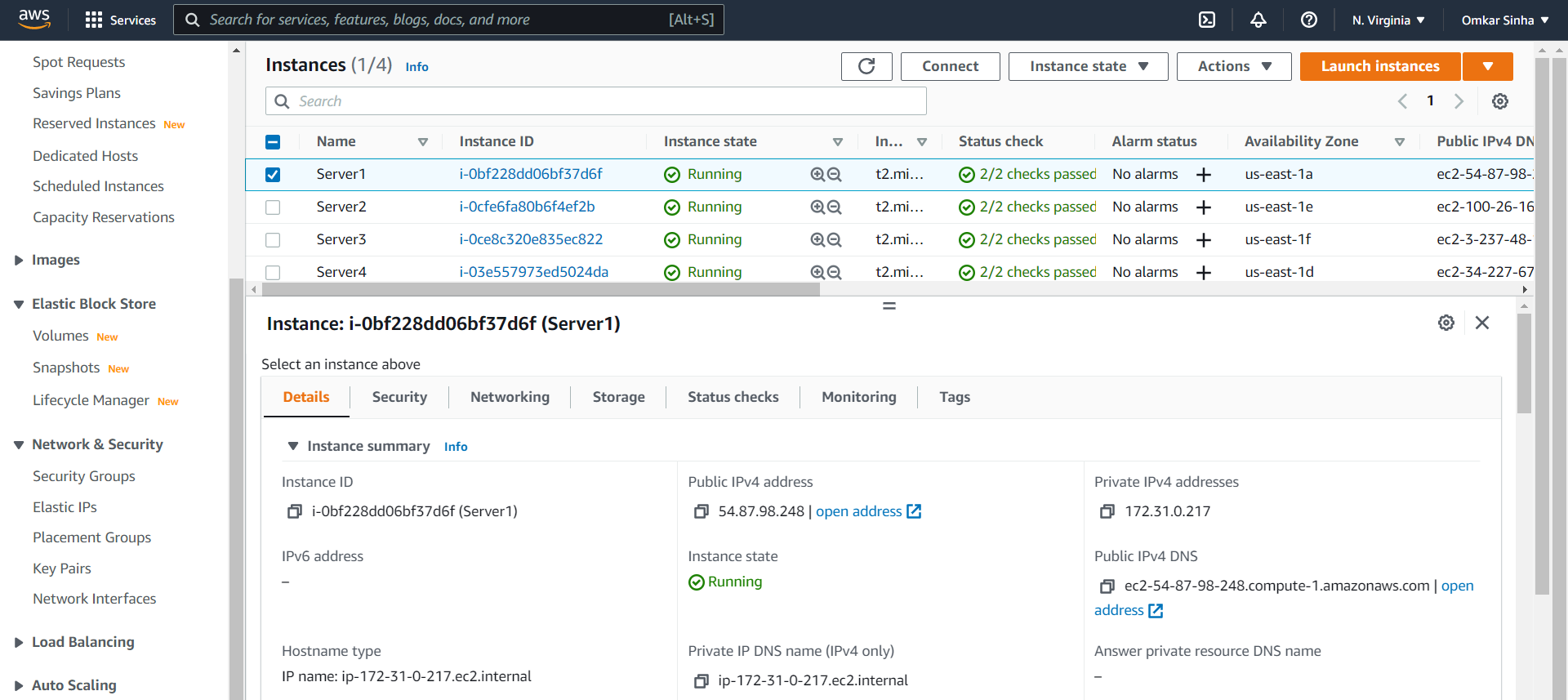
x) Now verifying that ec2 instances, security groups and network interfaces are successfully created in that order as follows respectively:



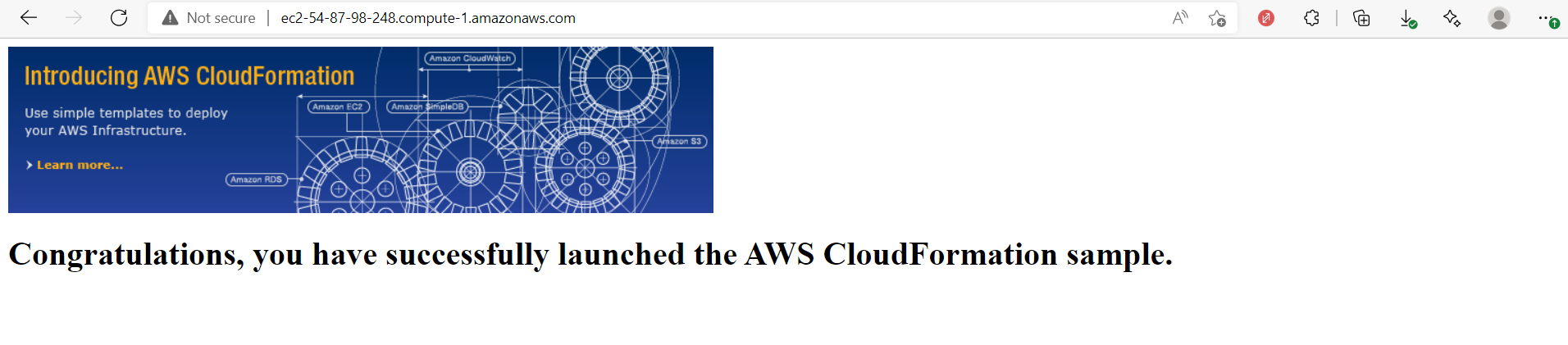




xi) We see the instance deatils as follows :



7) We visit the DNS link of each instance, and get following for each of them:

8) Now our goal is to change the index.html file for each of the servers so that we can verify the working of the load balancer.

i) We use putty to do this. We login to each instance specifying its IP-address, user-name(ec2-user) and then entering the private key(.ppk key created using puttygen), into putty and login.

ii) We then use following steps to modify the index.html:

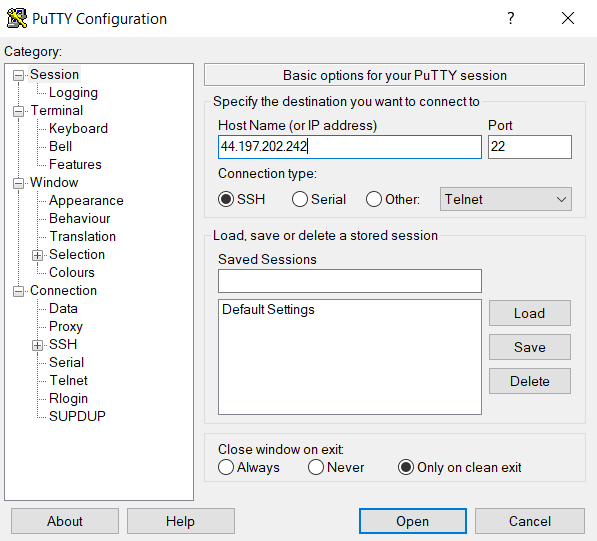
*cd /var/www*

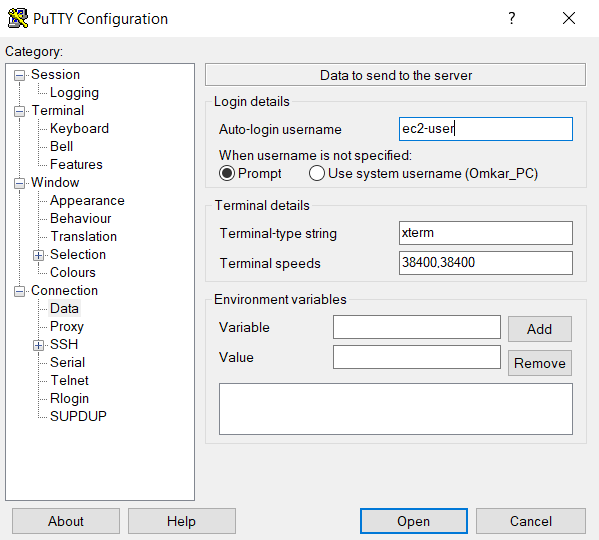
*cd html*

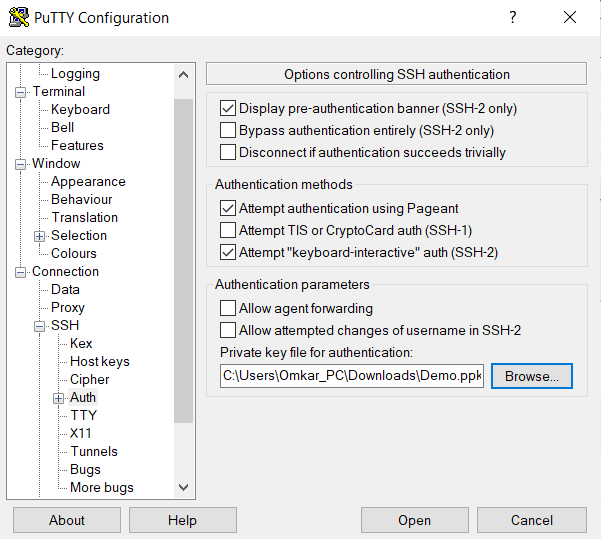
*sudo vim index.html*

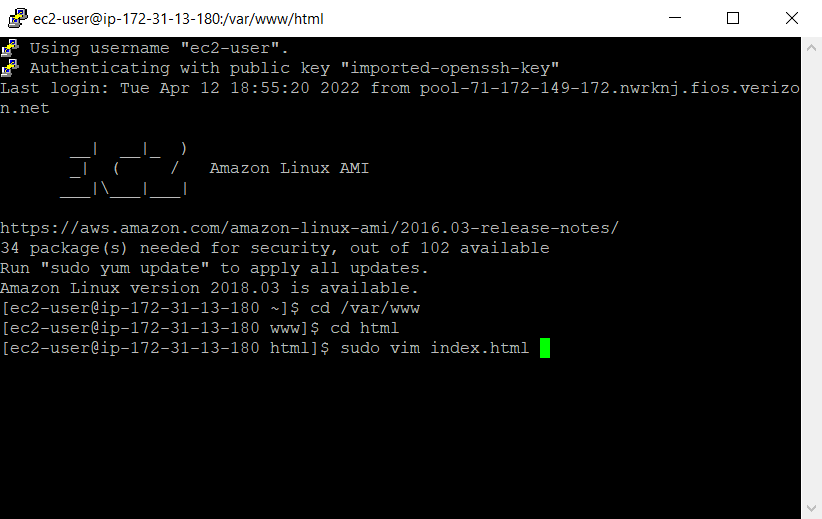
We change the index.html to include the Server names.

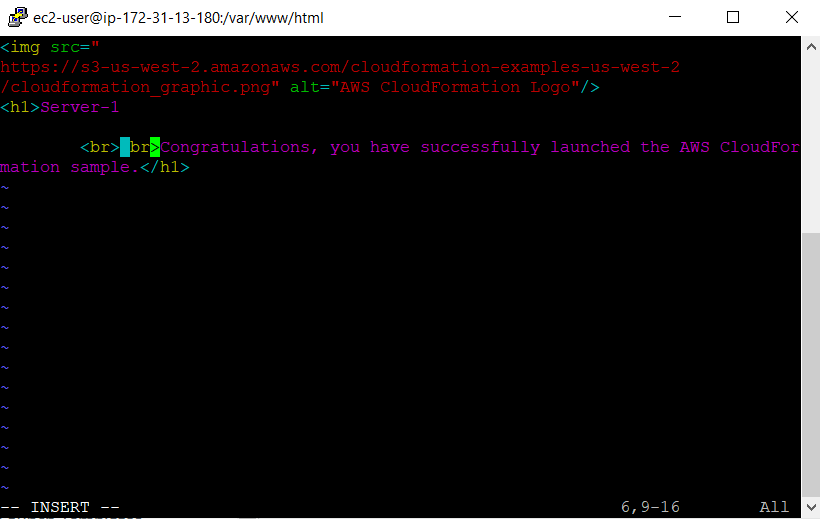
The screenshots for i) and ii)





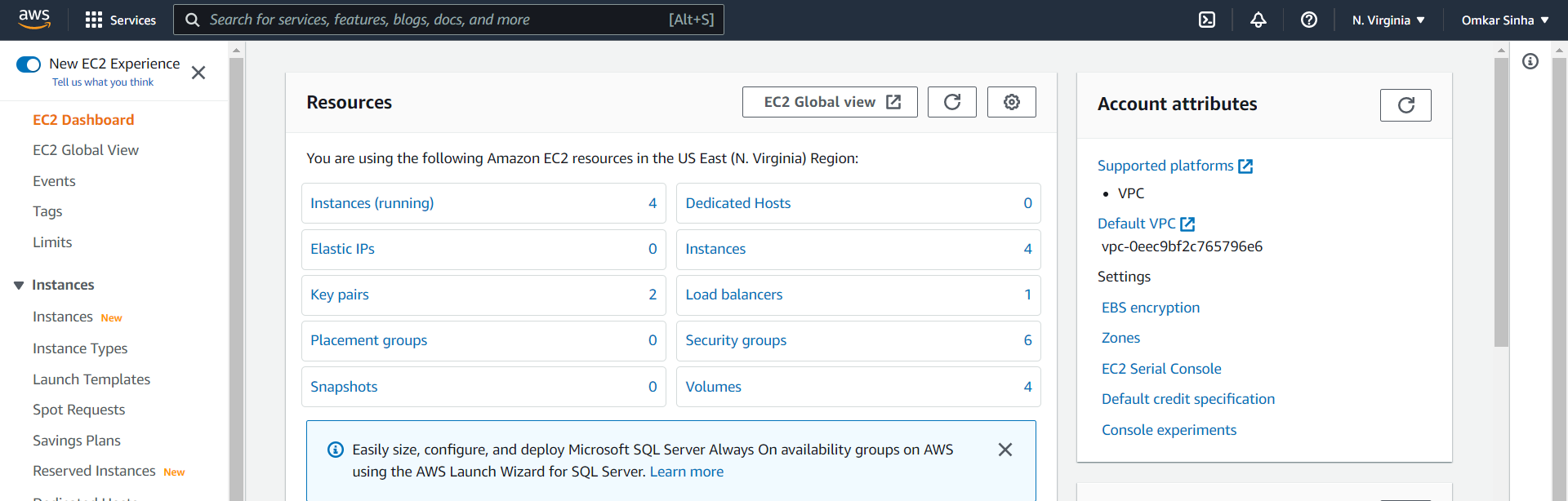




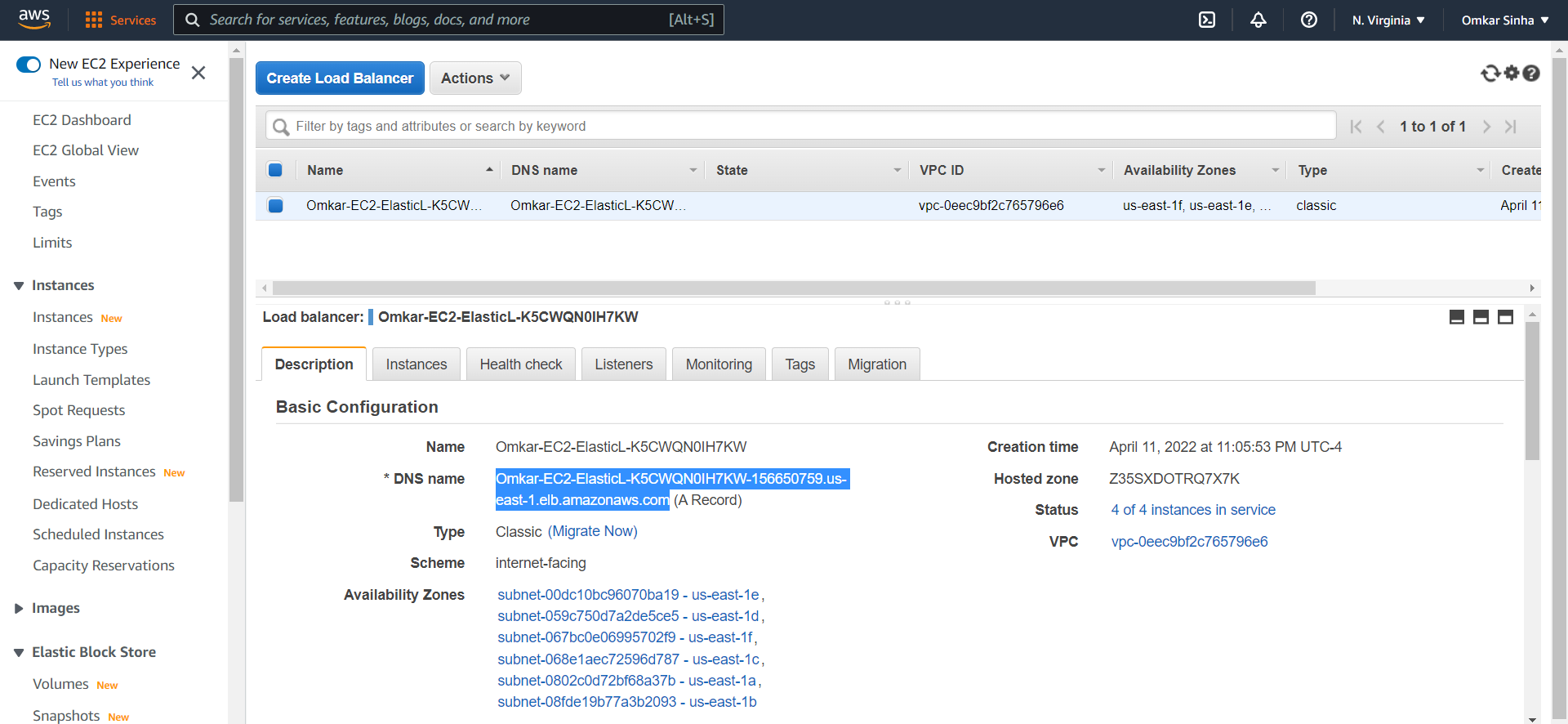


9) Now we verify the working of the load balancer:

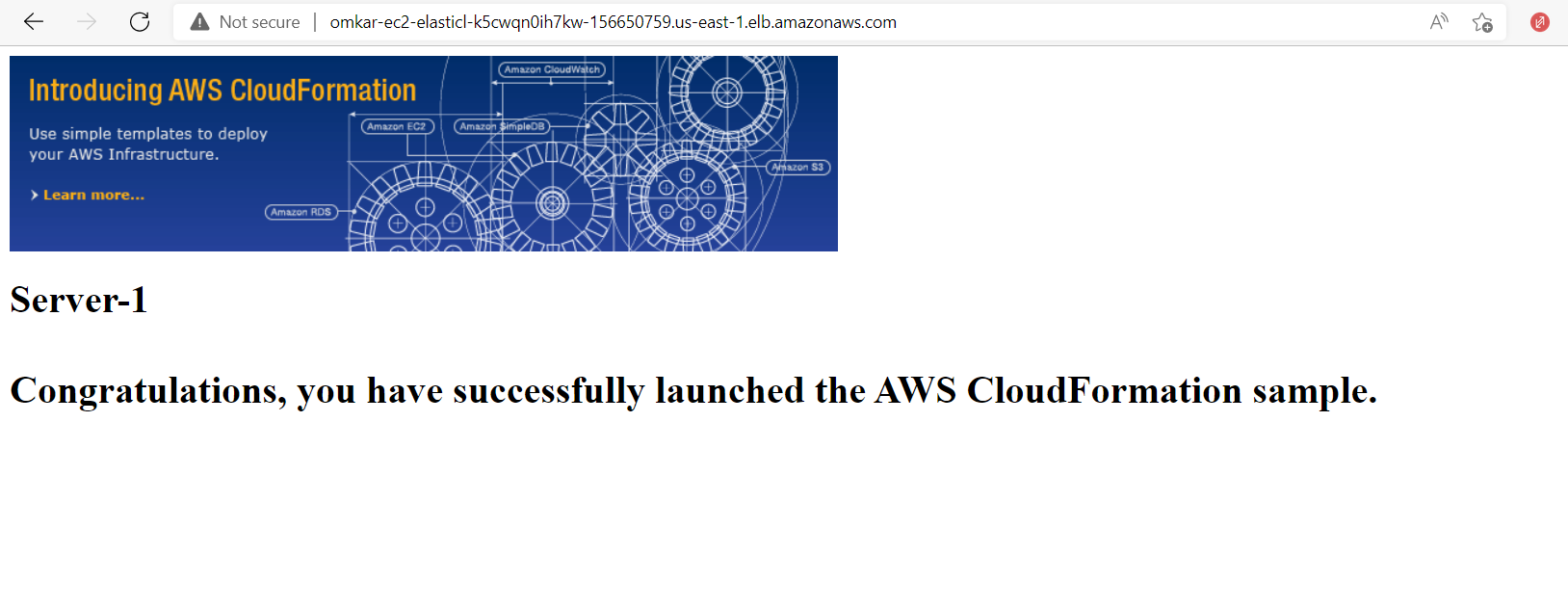
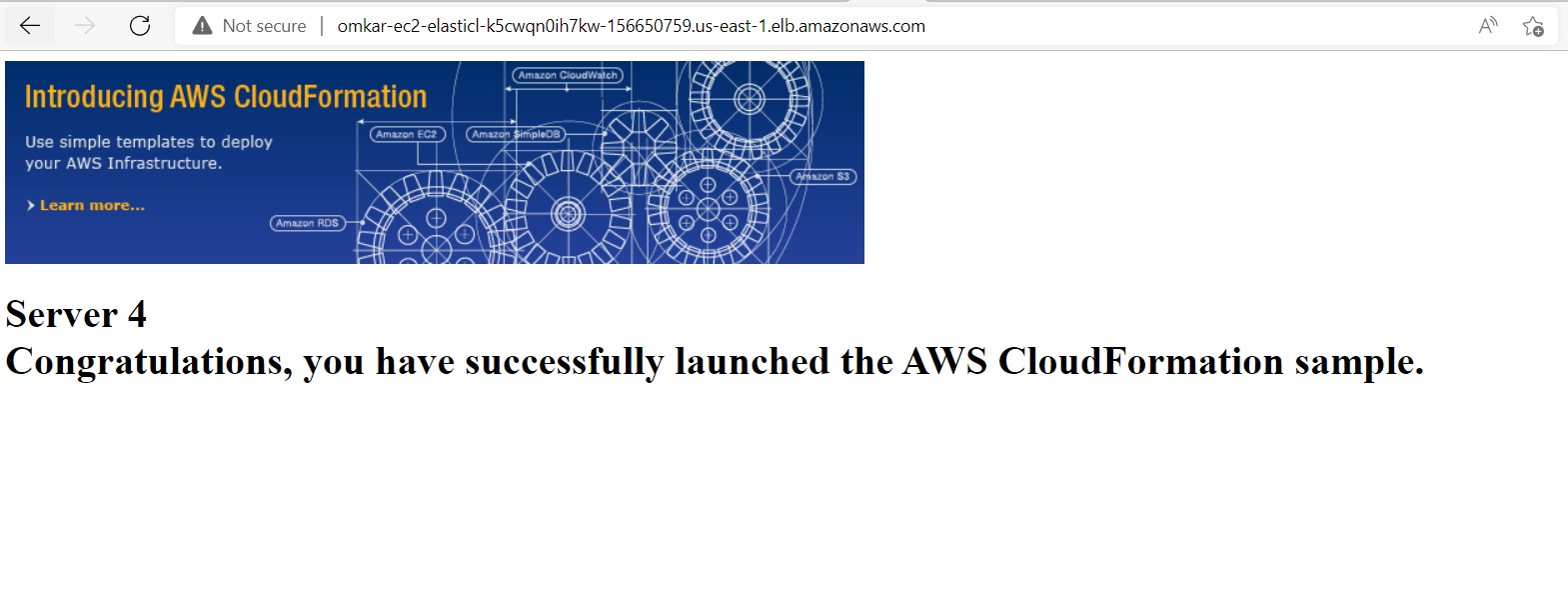
i) We first find the load balancer from “ec2 dashboard”

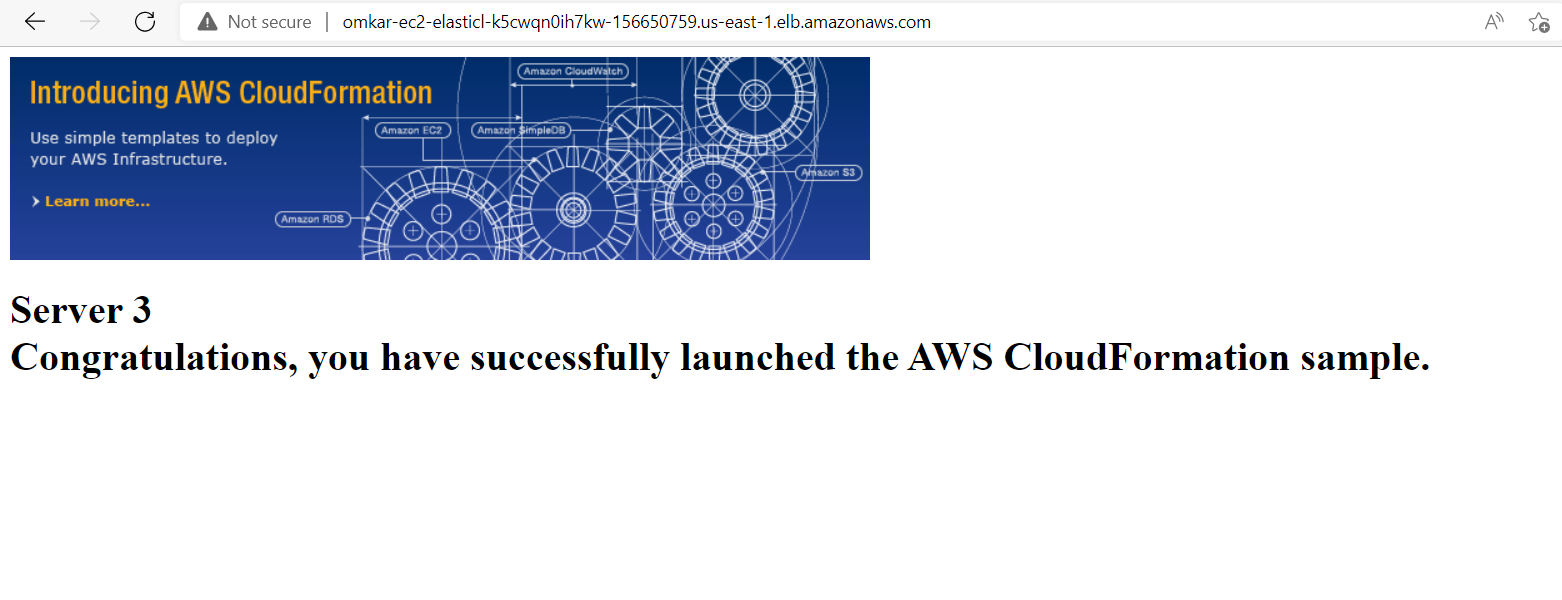


ii) We select the load balancer. We copy its DNS address:



10) We run the load balancer and get following output. The load balancer automatically balances the load and gives 4 different servers periodically.





I am getting the following order:- Server1, server4, server3, server2