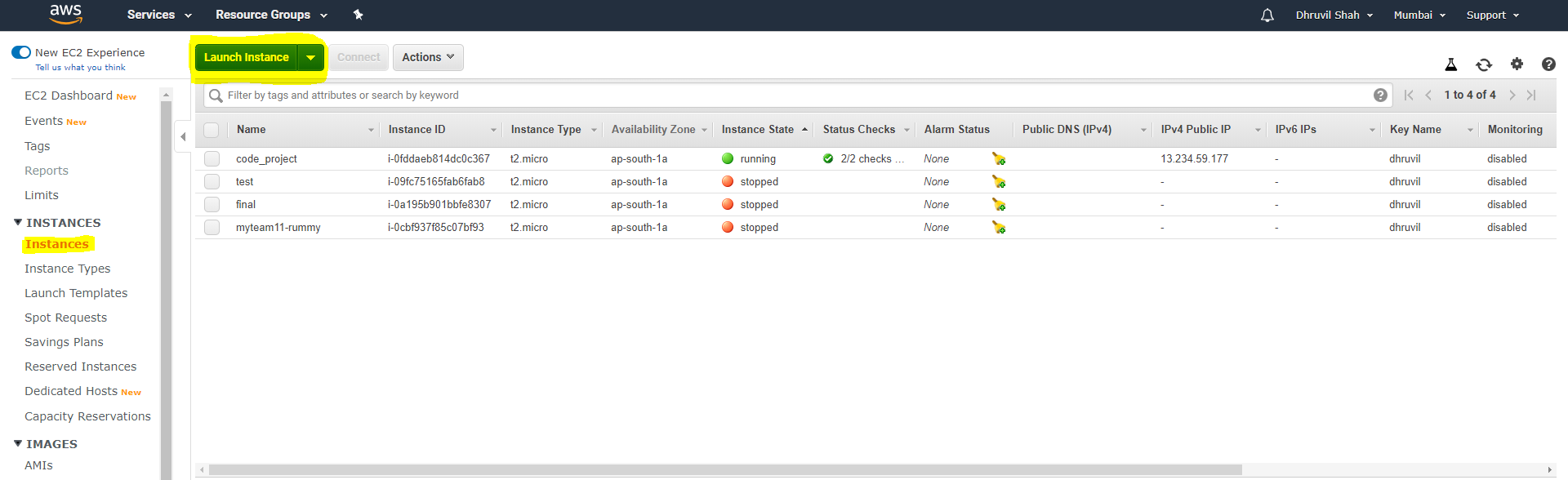
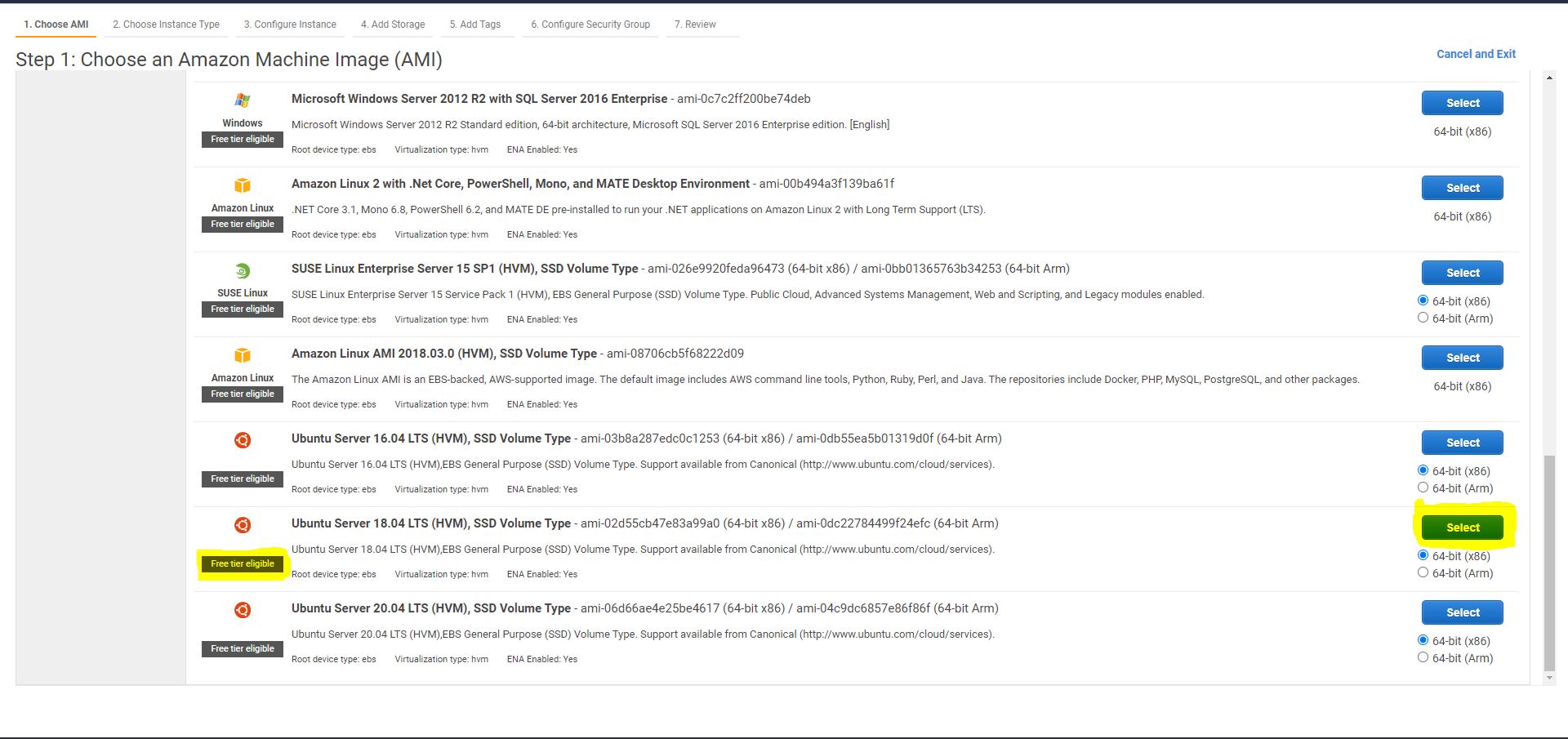
RabbitMQ Cluster Auto Scaling

* First you can create the EC2 instance in AWS(Name: *RMQ-Master*)

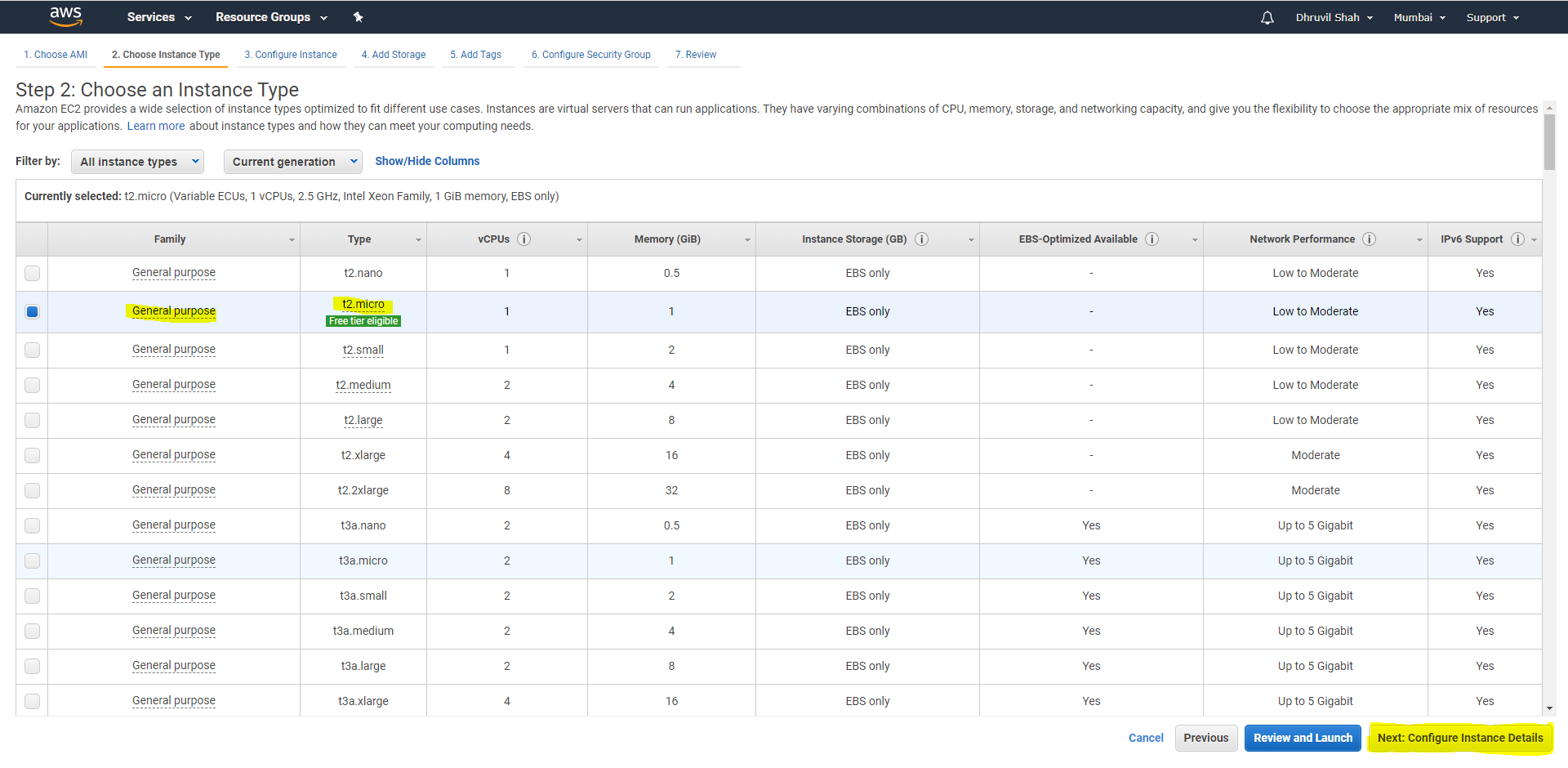


You can login the AWS-console and search the EC2.

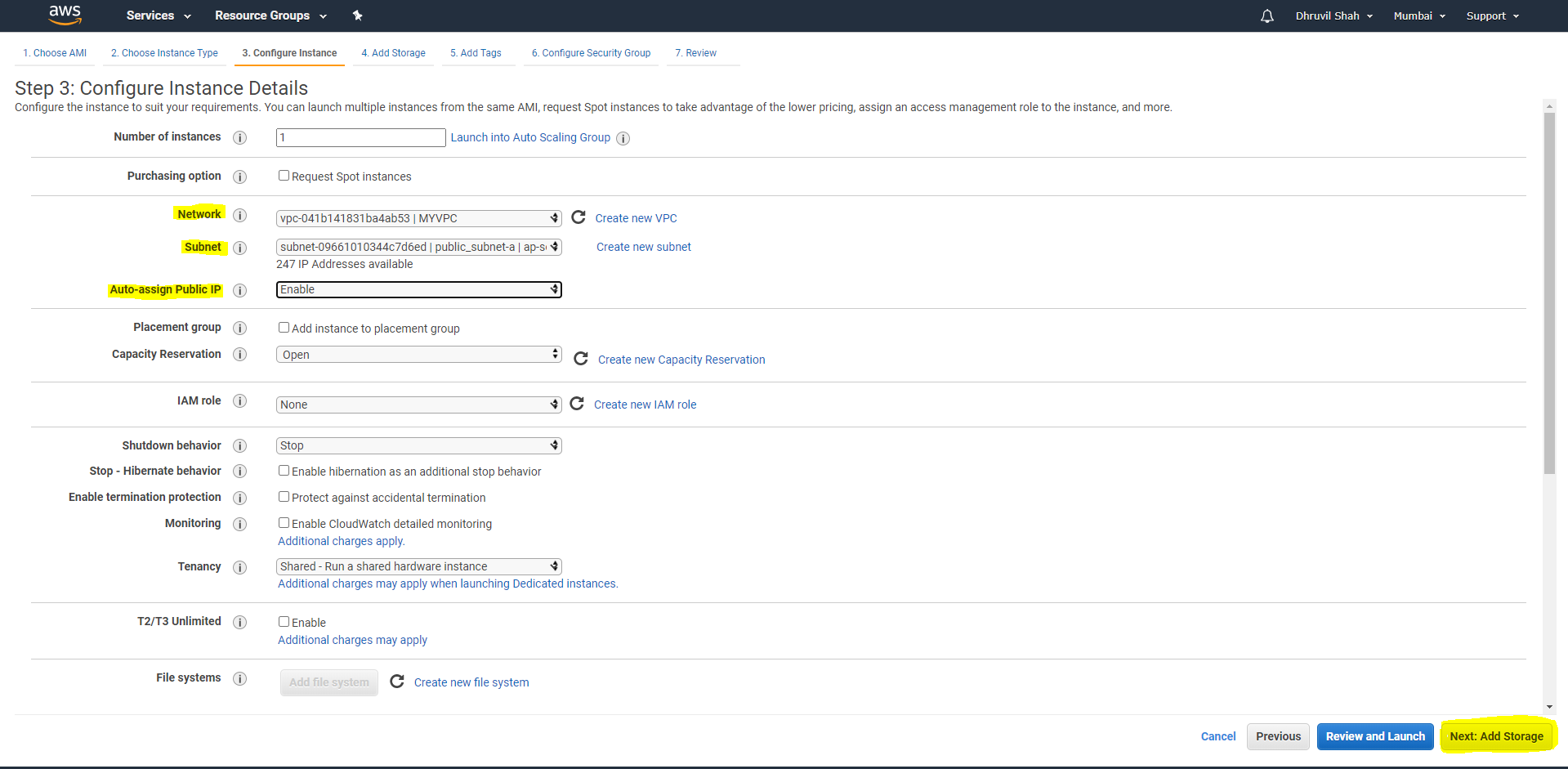
Click On Lunch Instance.



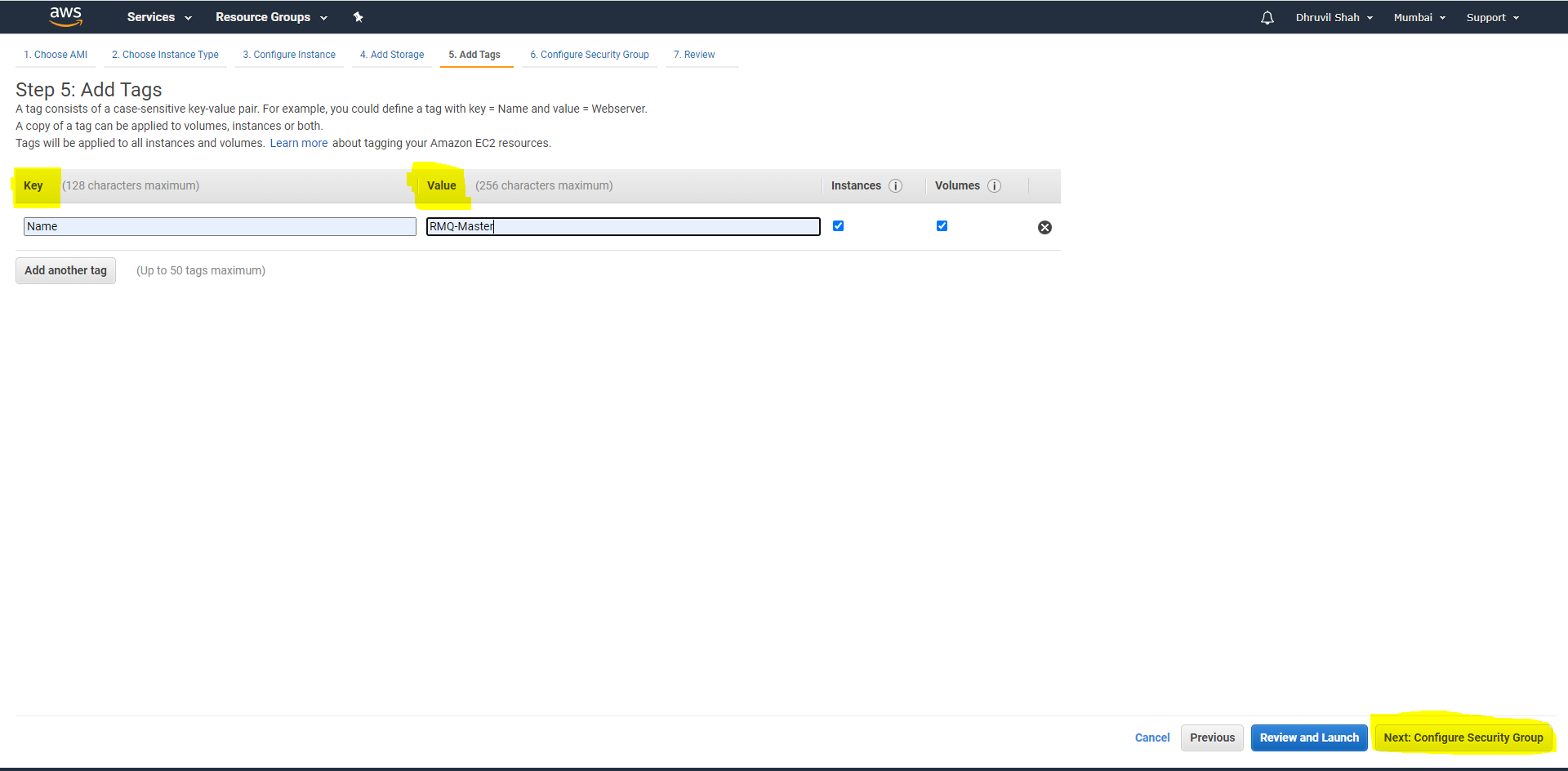
Select the free tire SSD volume of Ubuntu 18.04 LTS.



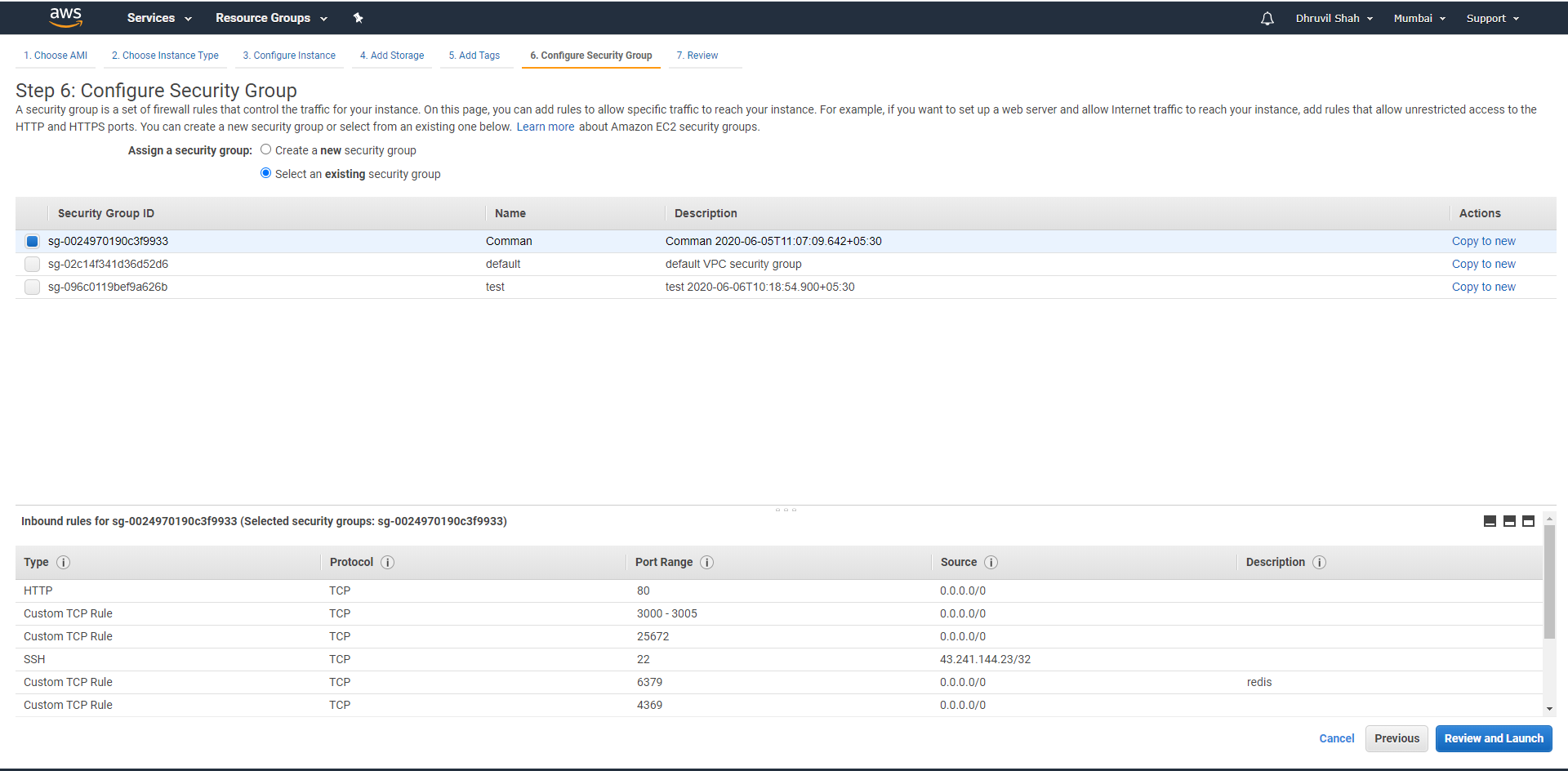
Select the t2micro and click on Next Button.



Select the Network, Subnet, and Auto-assign IP then click on Next Button.



Add the Key = Name and value = RMQ-Master and click on Next Button.



Whenever you cannot create the any security group then you can create the security group. And click review and launch button.

After lunch the instance. You can connect the instance whenever you can create.

Commands for the rabbitMQ install in the RabbitMQ-Master Instance.

* Install the ssh:

|  |
| --- |
| $ apt-get install ssh  $ apt-get update |

* Import Erlang GPG Key.

|  |
| --- |
| $ wget -O- https://packages.erlang-solutions.com/ubuntu/erlang\_solutions.asc | sudo apt-key add - |

* Add Erlang repository.

|  |
| --- |
| $ echo "deb https://packages.erlang-solutions.com/ubuntu bionic contrib" | sudo tee /etc/apt/sources.list.d/rabbitmq.list |

* Now Install Erlang package.

|  |
| --- |
| $ sudo apt update $ sudo apt -y install erlang |

* Let’s Install RabbitMQ now. Add RabbitMQ Keys.

|  |
| --- |
| $ wget -O- https://dl.bintray.com/rabbitmq/Keys/rabbitmq-release-signing-key.asc | sudo apt-key add - $ wget -O- https://www.rabbitmq.com/rabbitmq-release-signing-key.asc | sudo apt-key add - |

* Add RabbitMQ repository to ubuntu.

|  |
| --- |
| $ echo "deb https://dl.bintray.com/rabbitmq/debian bionic main" | sudo tee /etc/apt/sources.list.d/rabbitmq.list  $ apt-get update |

* Now Install RabbitMQ server.

|  |
| --- |
| $ sudo apt install rabbitmq-server |

* Enable the rabbitmq-server.

|  |
| --- |
| $ systemctl enable rabbitmq-server  $ systemctl start rabbitmq-server |

* rabbitMQ plugin enable.

|  |
| --- |
| $ rabbitmq-plugins enable rabbitmq\_management |

* Create RabbitMQ user and provide full permissions.

|  |
| --- |
| $ rabbitmqctl add\_user test test $ rabbitmqctl set\_user\_tags test administrator $ rabbitmqctl set\_permissions -p / test ".\*" ".\*" ".\*" |

* Restart the rabbitmq-server service.

|  |
| --- |
| $ systemctl restart rabbitmq-server.service  $ systemctl status rabbitmq-server.service  $ rabbitmqctl status  $ sudo rabbitmq-plugins list |

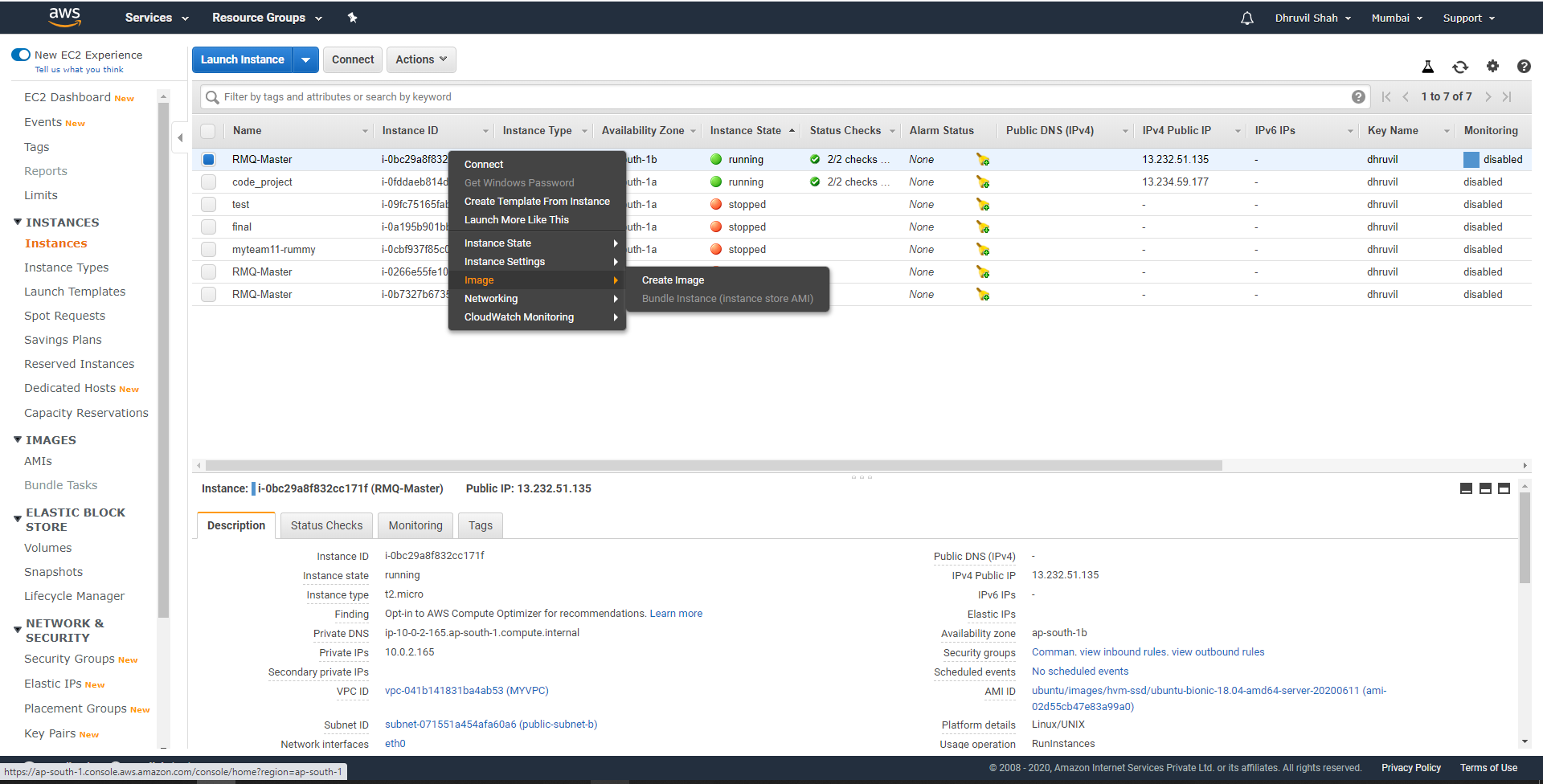
* Add the policies.

|  |
| --- |
| $ rabbitmqctl set\_policy ha-all ".\*" '{"ha-mode":"all"}' |

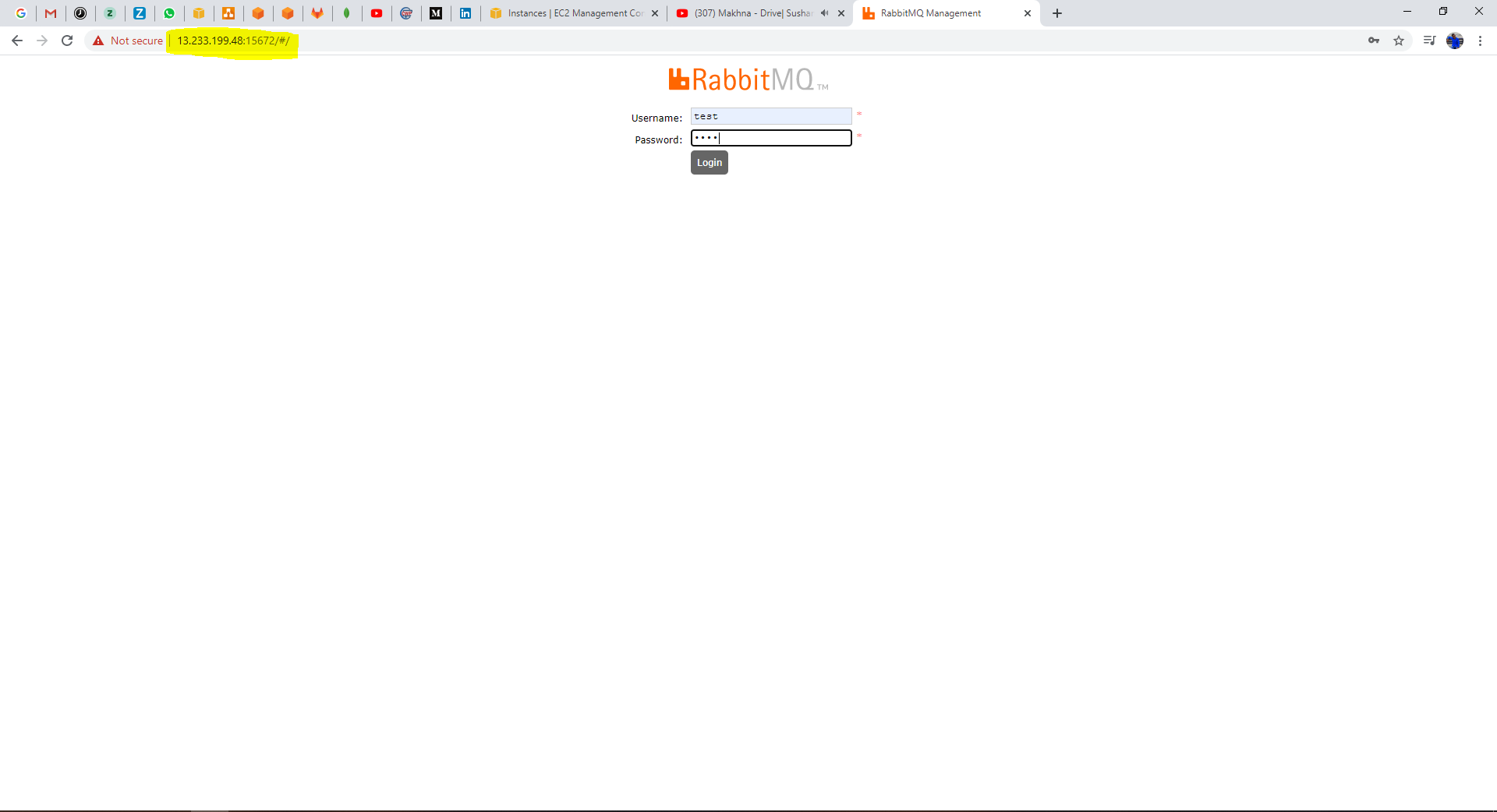
* Python install.

|  |
| --- |
| $ apt-get install python |

* Create the image of RabbitMQ-master instance.

****

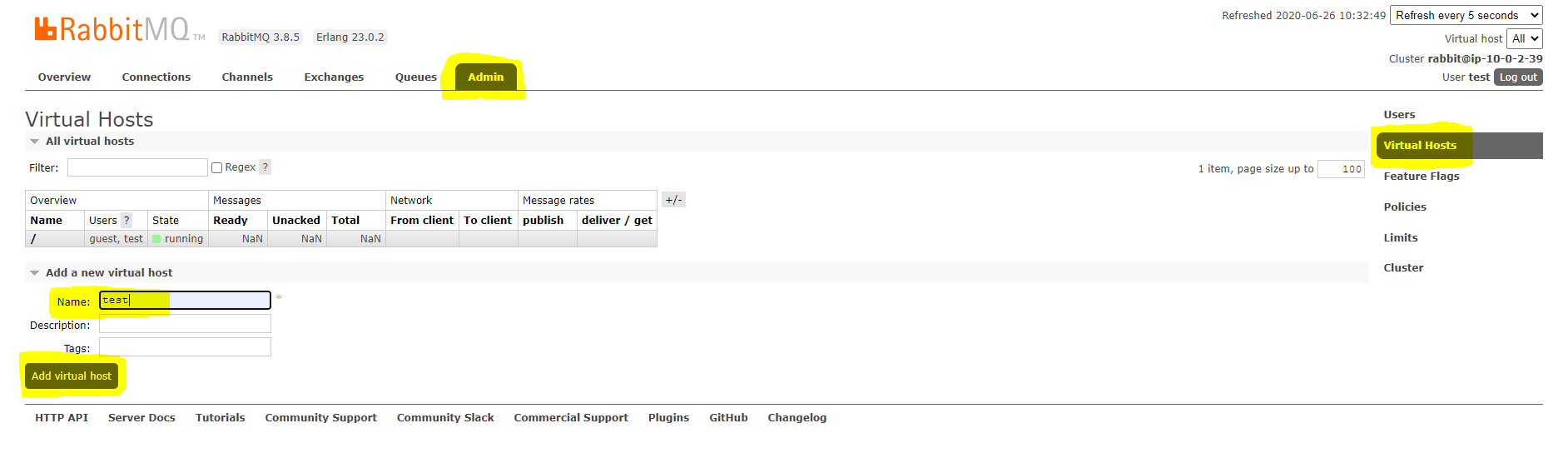
* Now the login the rabbimq server.



You can login via your EC2 instance Public IP

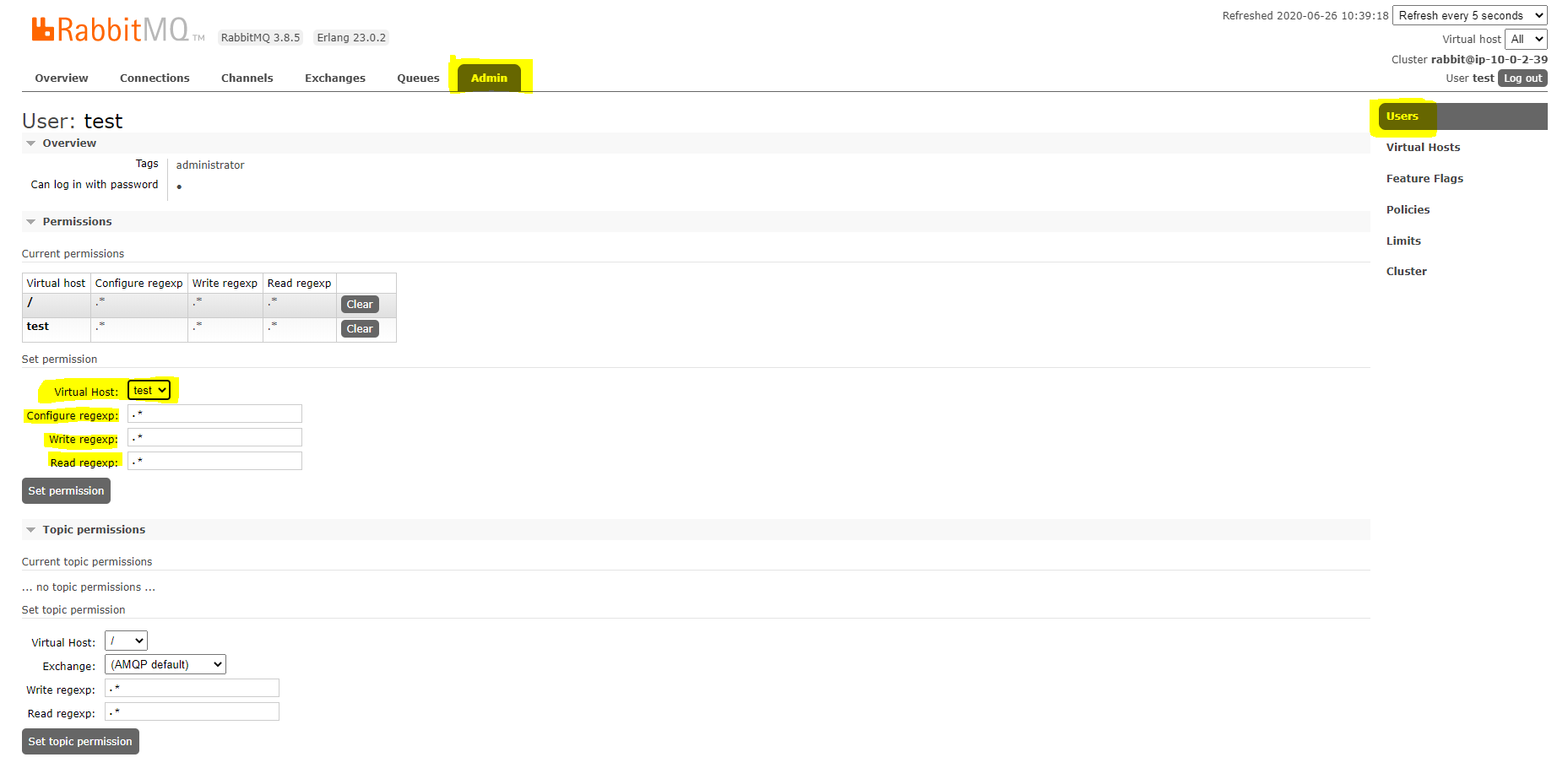
<http://public_ip:15672/>

* Create the virtual host.



You can create the virtual host in server. And define the name and click the **Add Virtual** Host button.

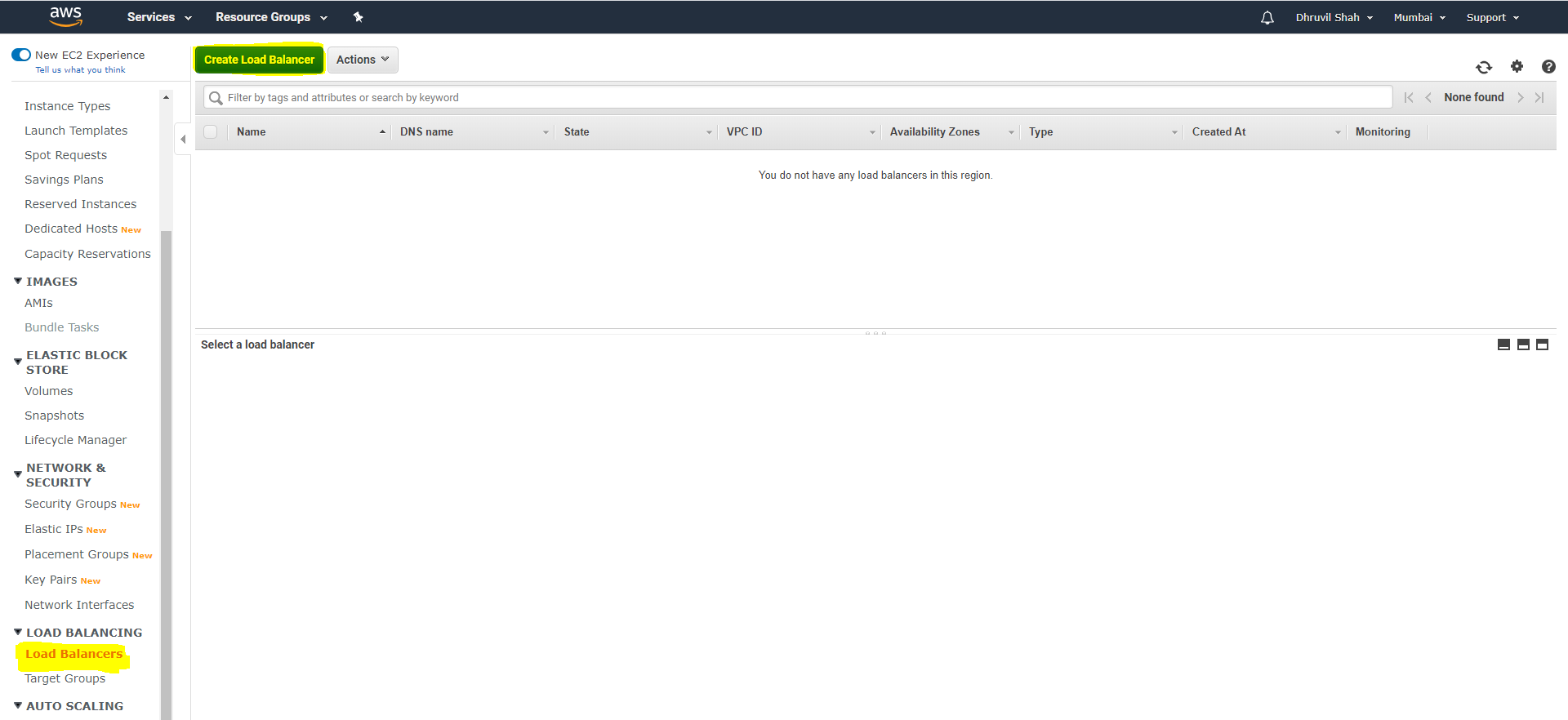
* Assign the permission of user for the virtual host.



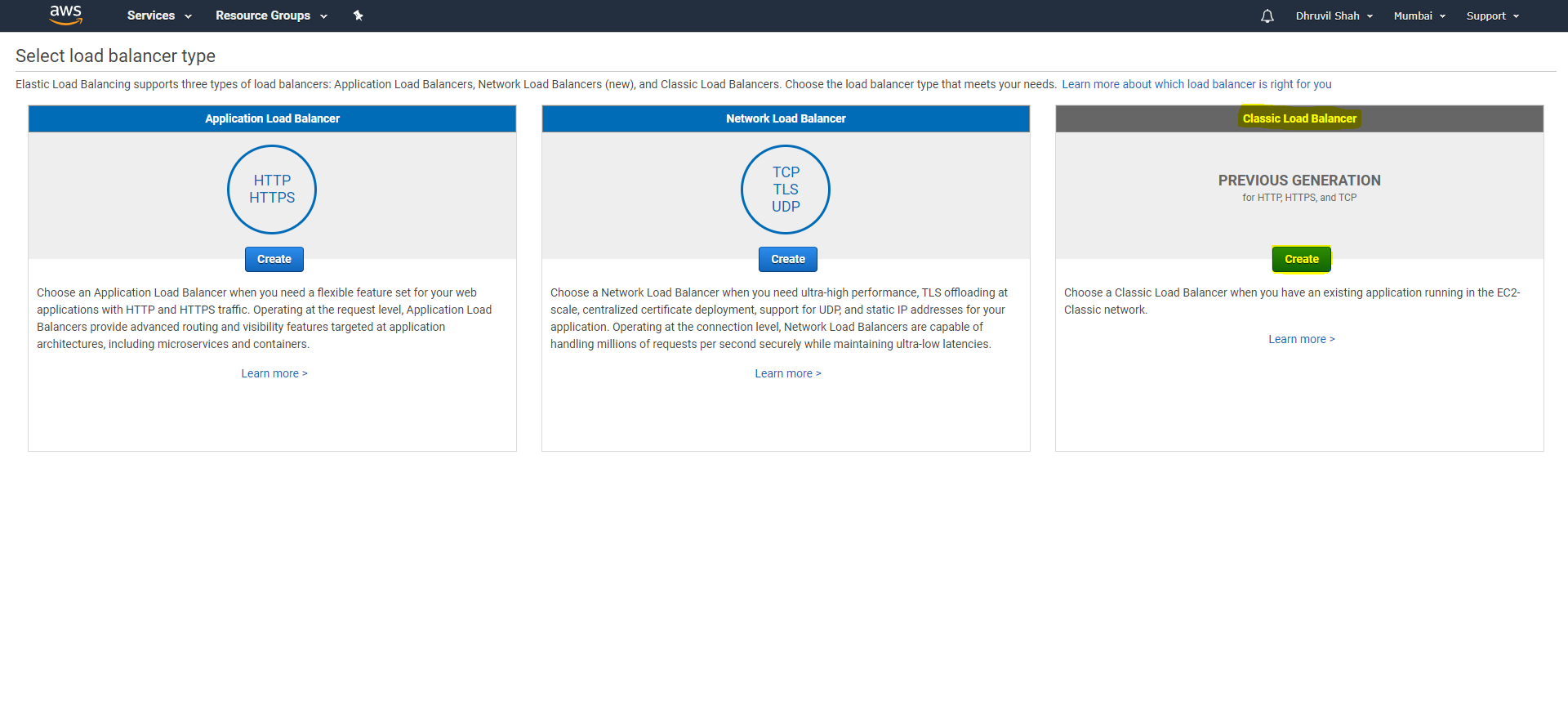
Whenever we create the virtual host and it will be assign the user(test).

Create the Load Balancer.

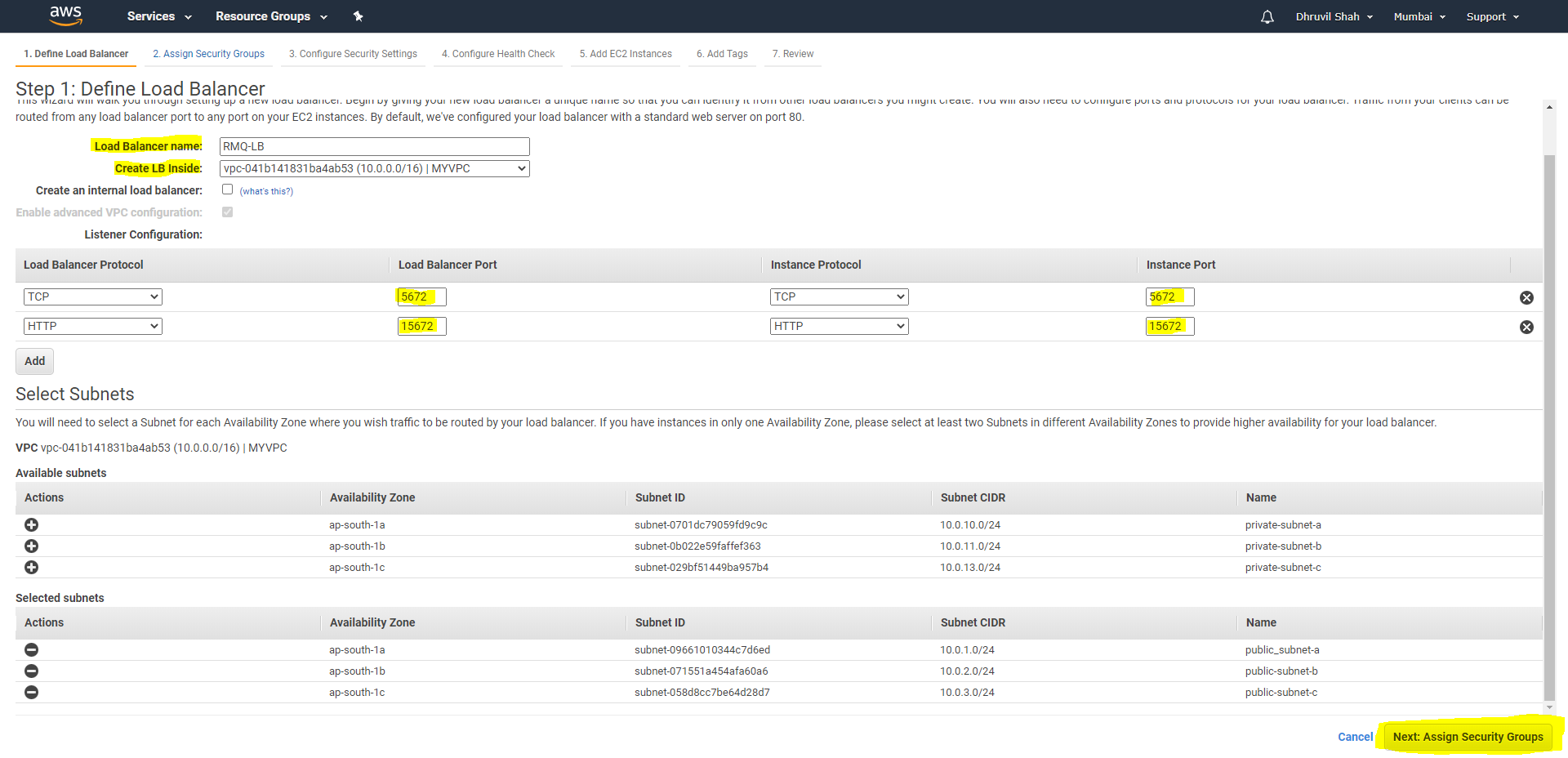
* Create the load Balancer.



* Create the classic load balancer.

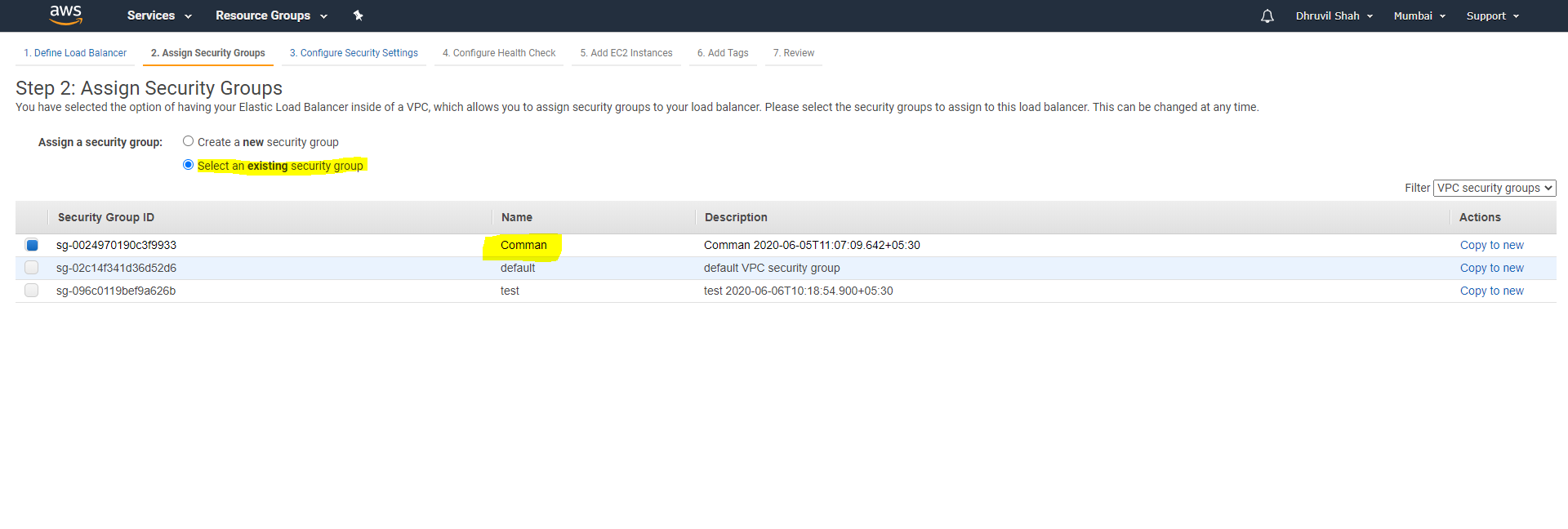


* Port define.



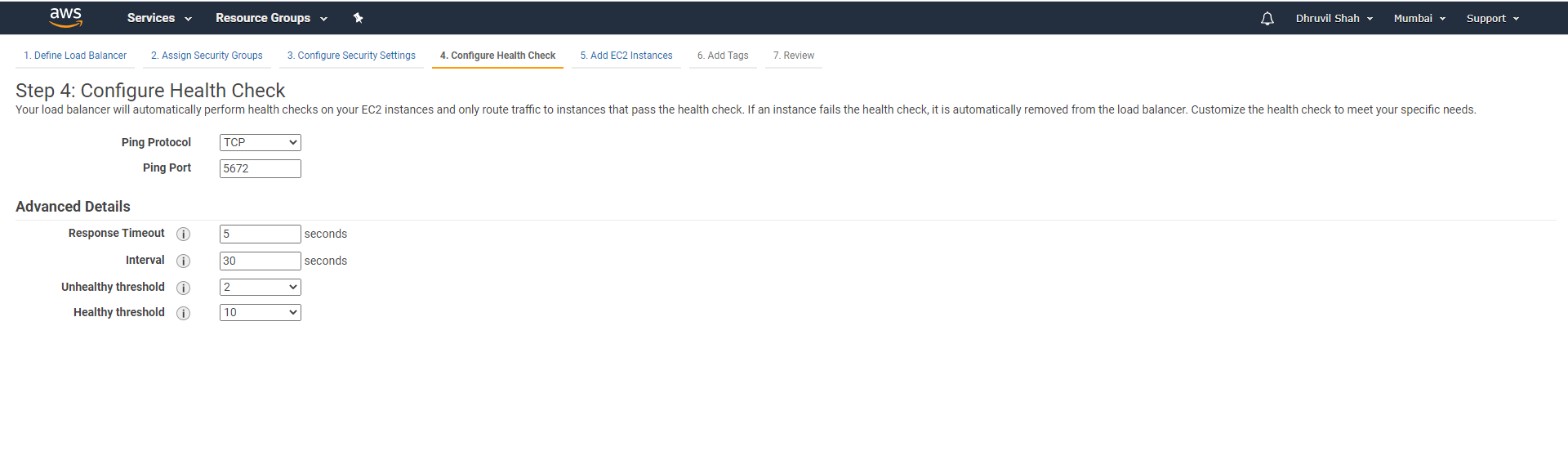
Define the Load Balancer name and port.

* Assign Security Groups.

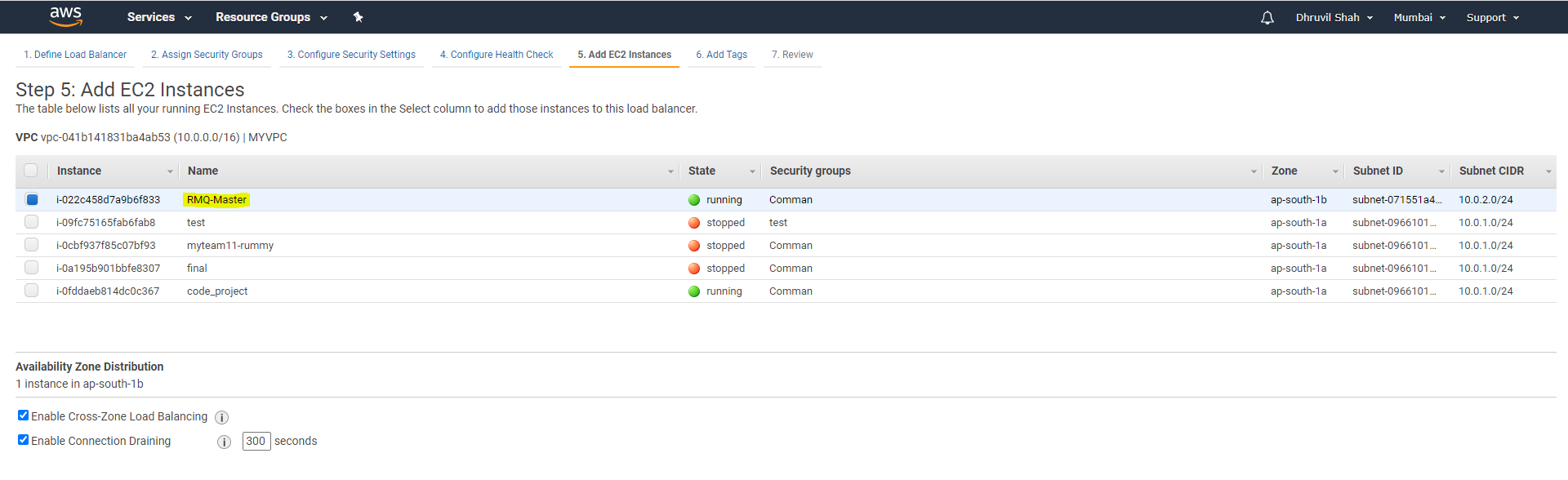


Make the comman security group and assign the port number like (15672,5672)

* Configure Health Check.

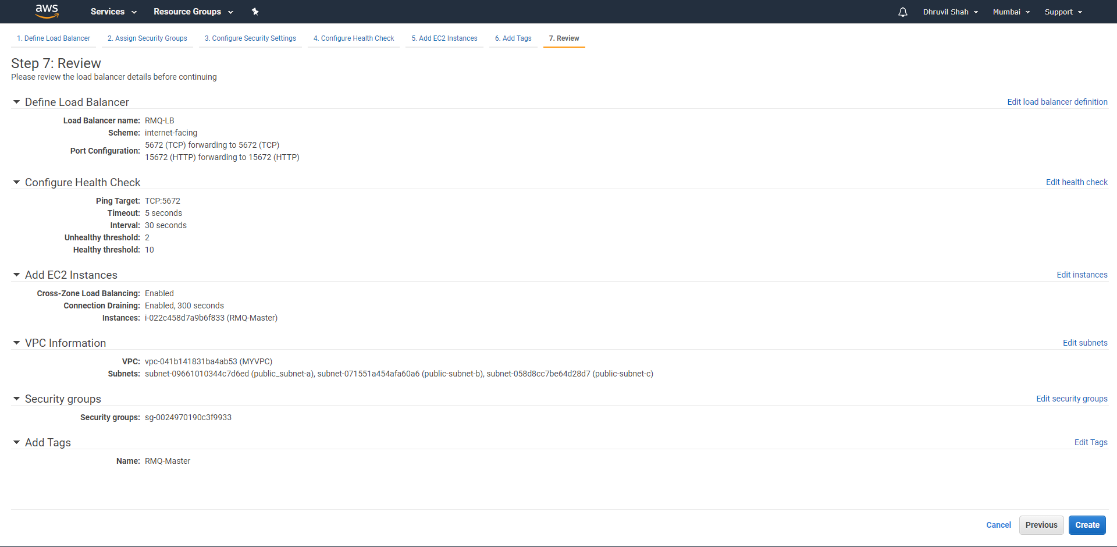


* Add EC2 Instances.



Add the RMQ-Master Instance in to Load Balancer.

* Create the Load Balancer.

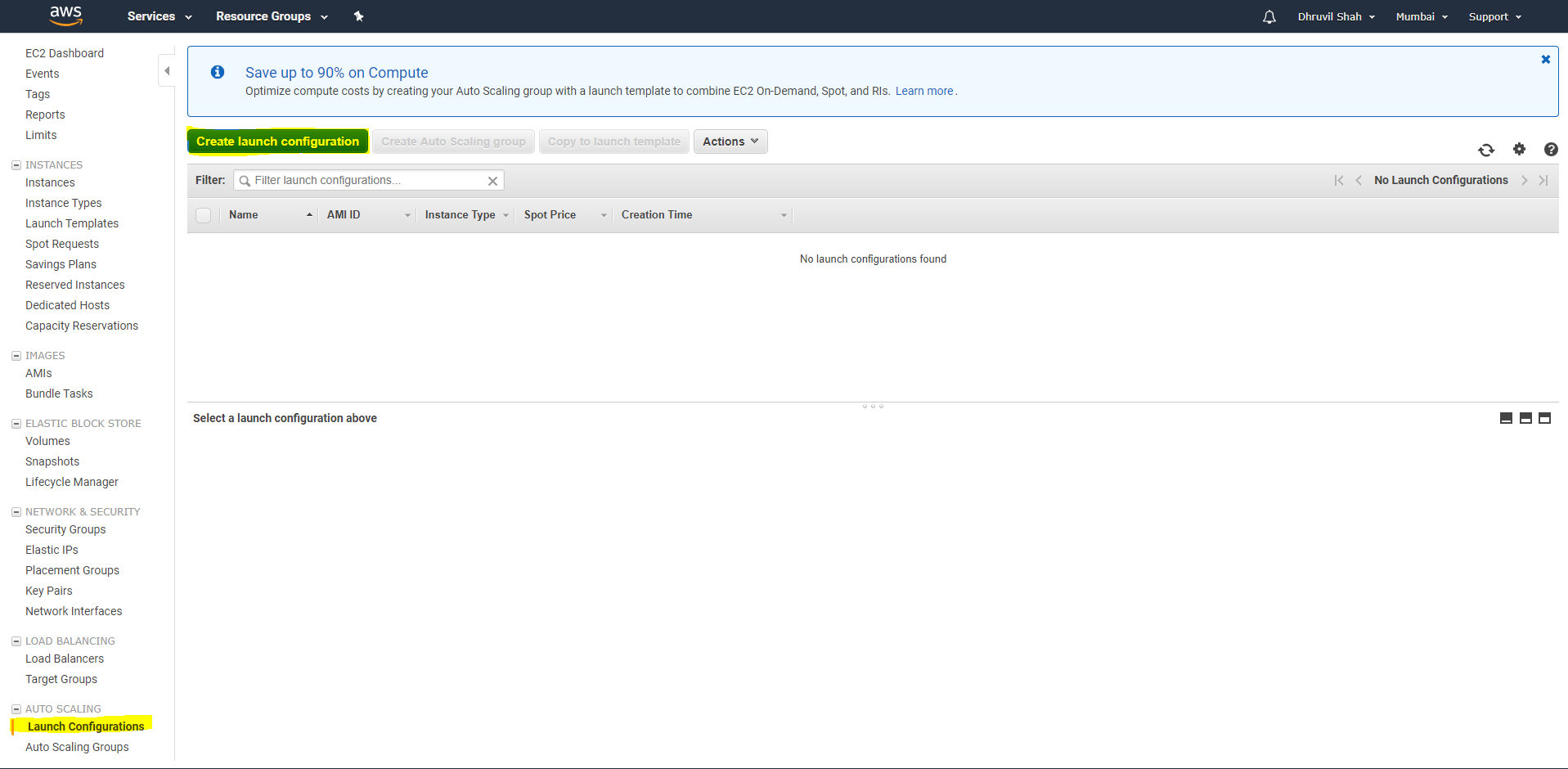


* Set the permission for the particular user. Whenever we create.(for test user)

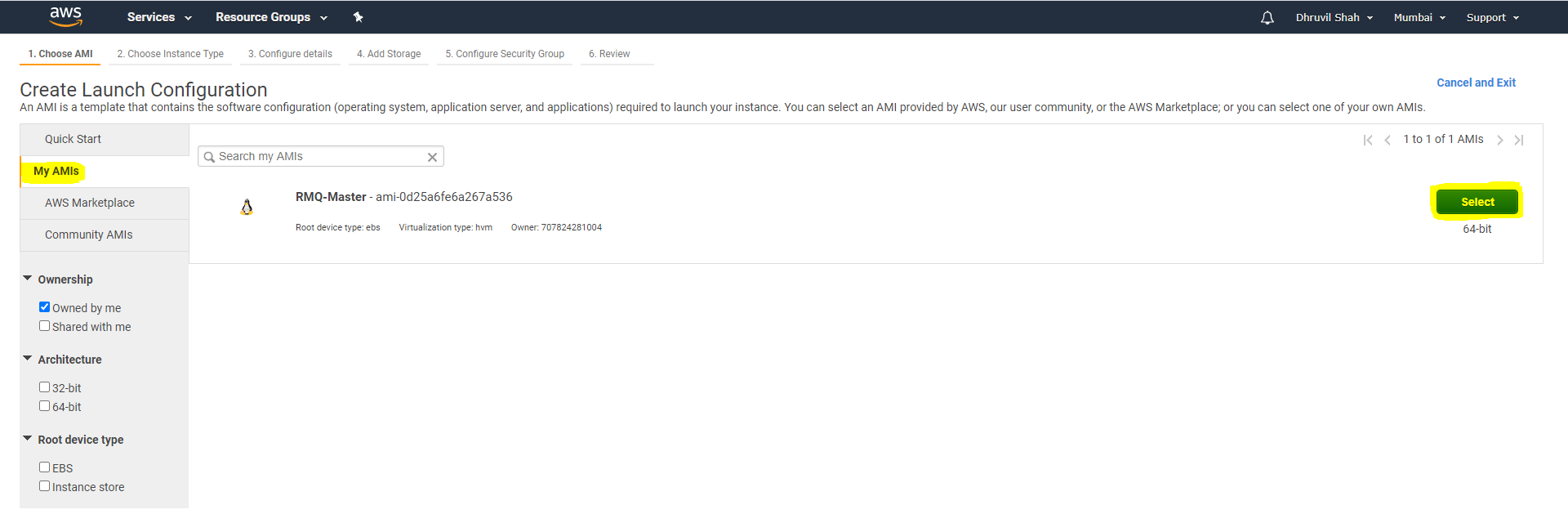
|  |
| --- |
| $ rabbitmqctl set\_policy -p test ha-all "" '{"ha-mode":"all","ha-sync-mode":"automatic","ha-promote-on-shutdown":"always"}' |

Auto Scaling.

* Create the Lunch Configuration.

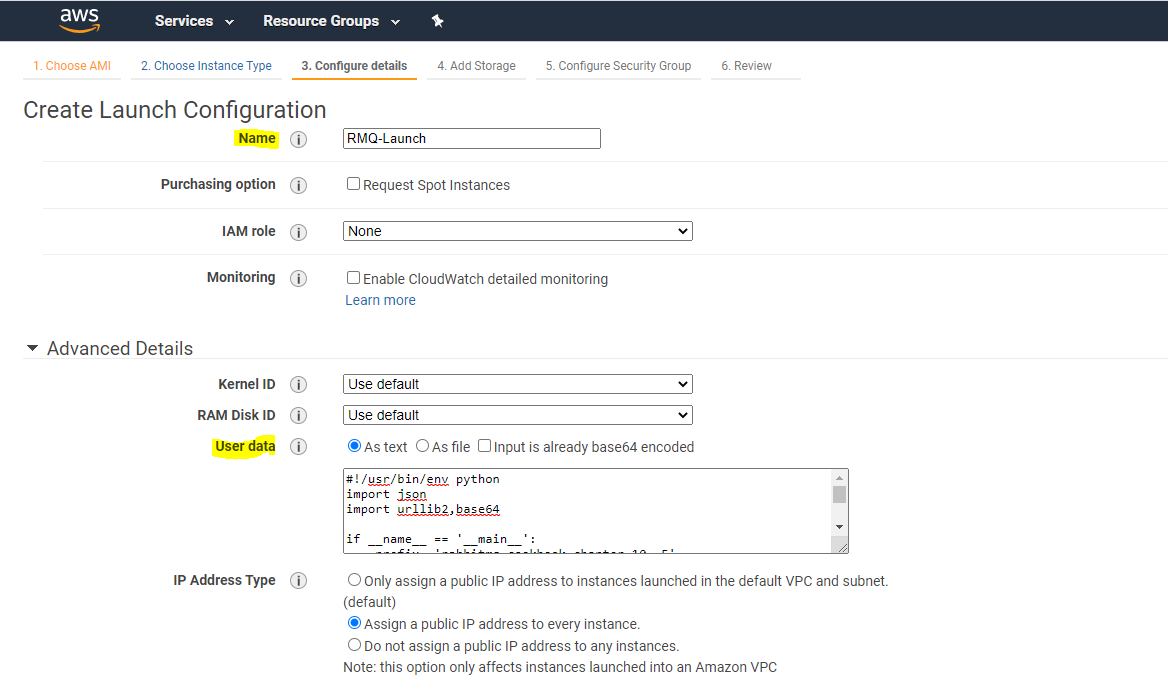


* Select the AIM image for the launch configuration.



Select the RMQ-Master AIM for the Lunch Configuration.

* Create Launch Configuration.

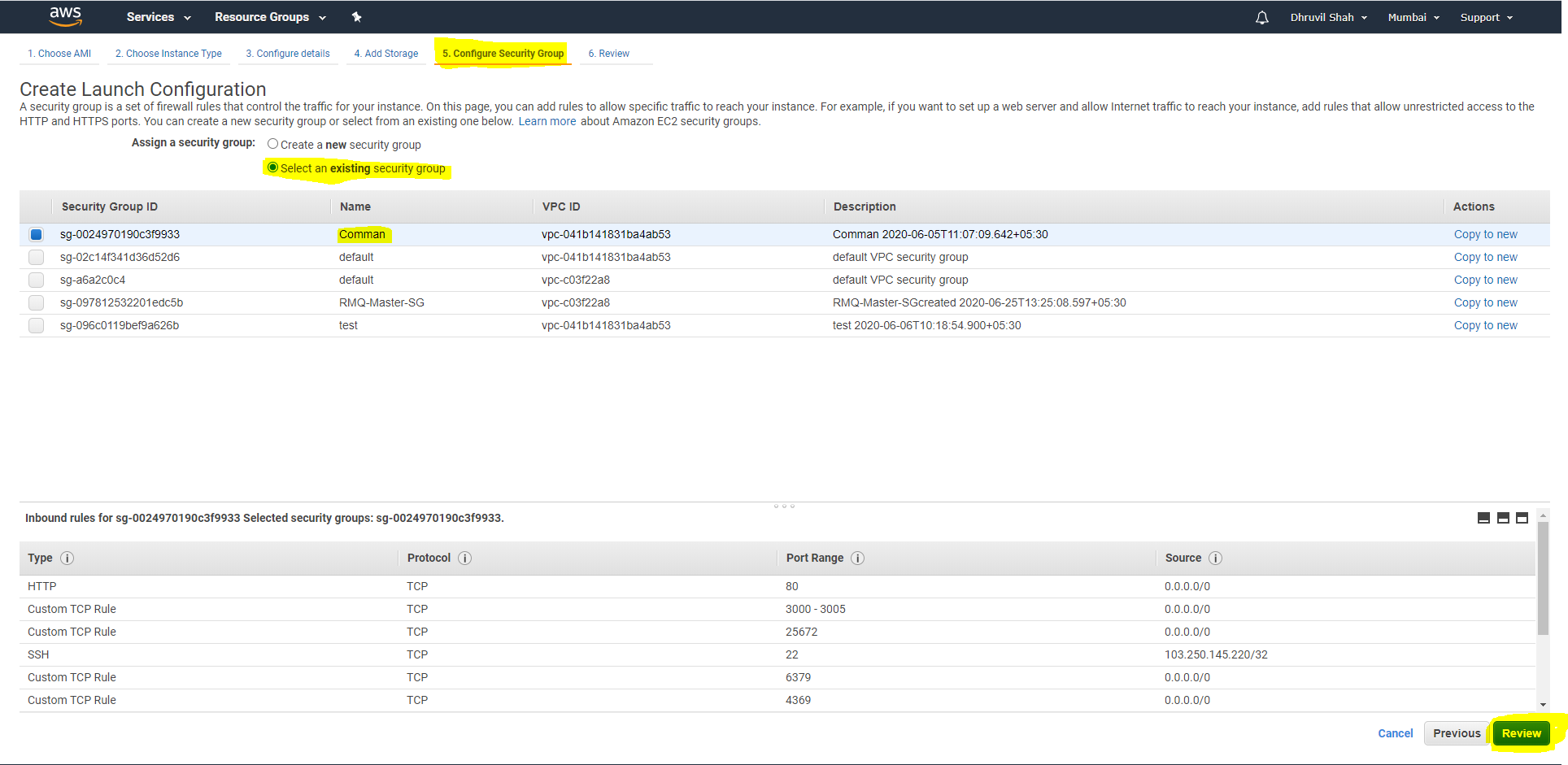


|  |
| --- |
| #!/usr/bin/env python  import json  import urllib2,base64  if \_\_name\_\_ == '\_\_main\_\_':  prefix ='rabbitmq cookbook chapter 10 .5'  from subprocess import call  call(["rabbitmqctl", "stop\_app"])  call(["rabbitmqctl", "reset"])  try:  \_url = 'http://RMQ-LB-863402881.ap-south-1.elb.amazonaws.com:15672/api/nodes'  print prefix + 'Get json info from ..' + \_url  request = urllib2.Request(\_url)  base64string = base64.encodestring('%s:%s' % ('test', 'test')).replace('\n', '')  request.add\_header("Authorization", "Basic %s" % base64string)  data = json.load(urllib2.urlopen(request))  print prefix + 'request ok... finding for running node'    for r in data:  if r.get('running'):  print prefix + 'found running node to bind..'  print prefix + 'node name: '+ r.get('name') +'- running:' + str(r.get('running'))  from subprocess import call  call(["rabbitmqctl", "join\_cluster",r.get('name')])  break;  pass  except Exception, e:  print prefix + 'error during add node'  finally:  from subprocess import call  call(["rabbitmqctl", "start\_app"])  pass |

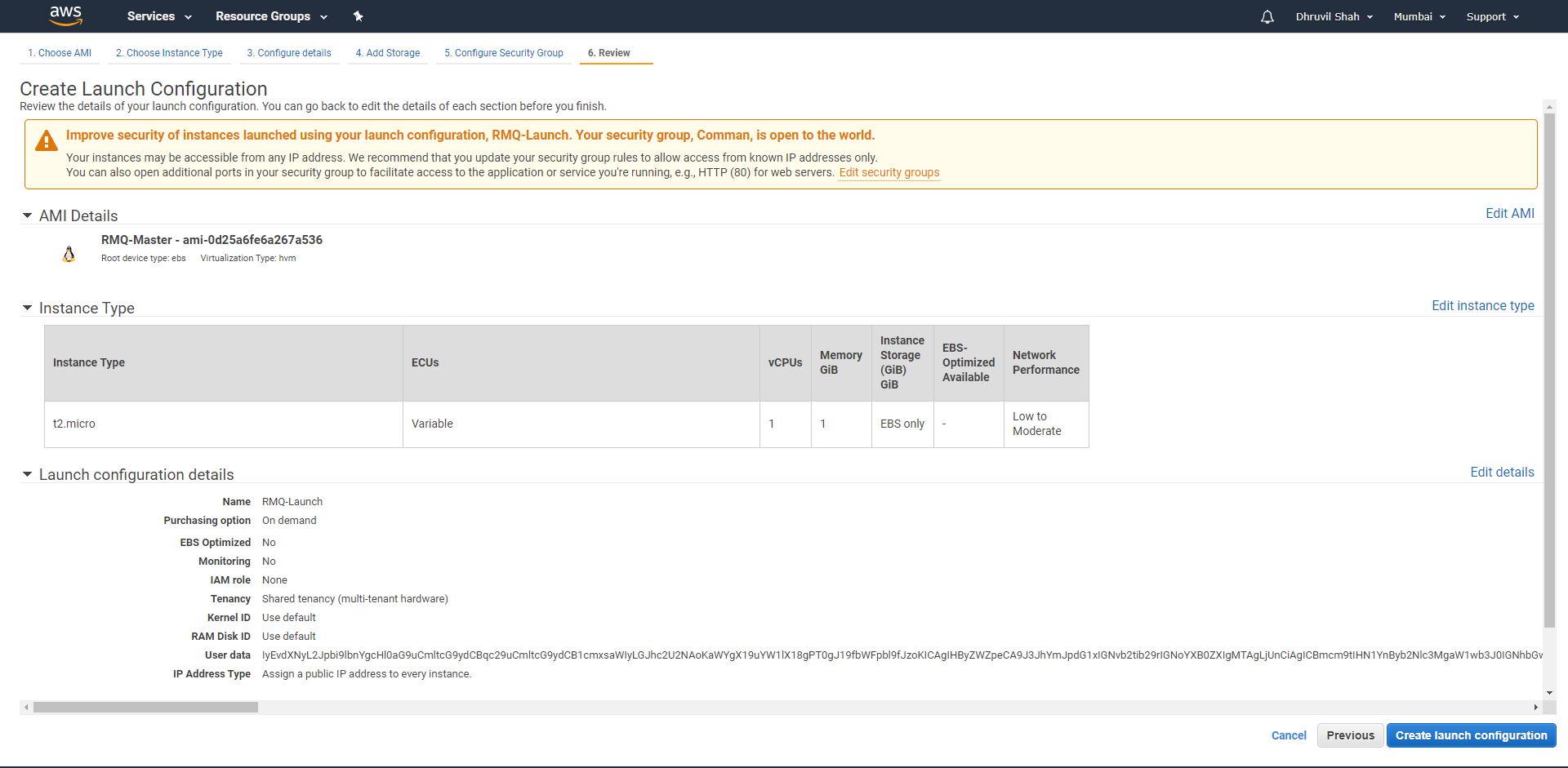
Add the name of Launch configuration. Add the Script.

Change the URL for the load balancer. And Username and Password.

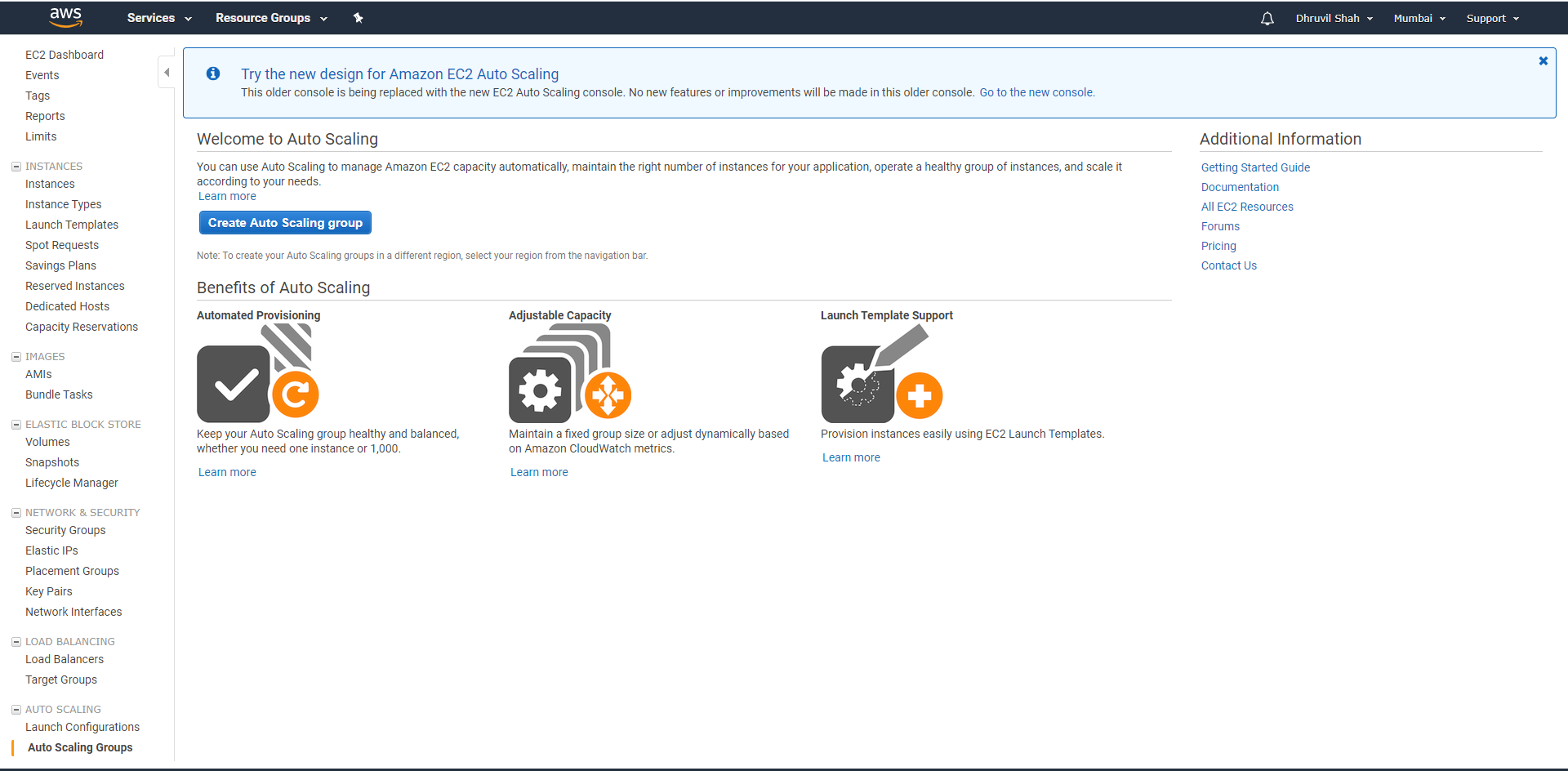
* Configure the security group.



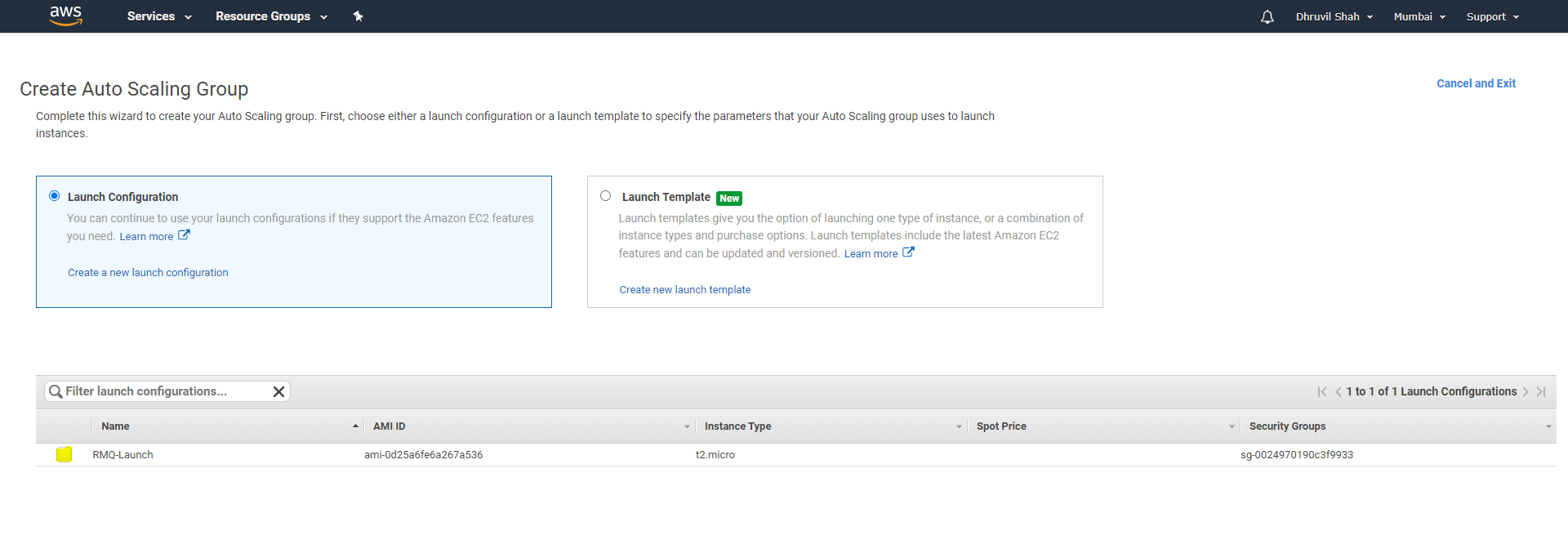
* Review and the create the Lunch Configuration.



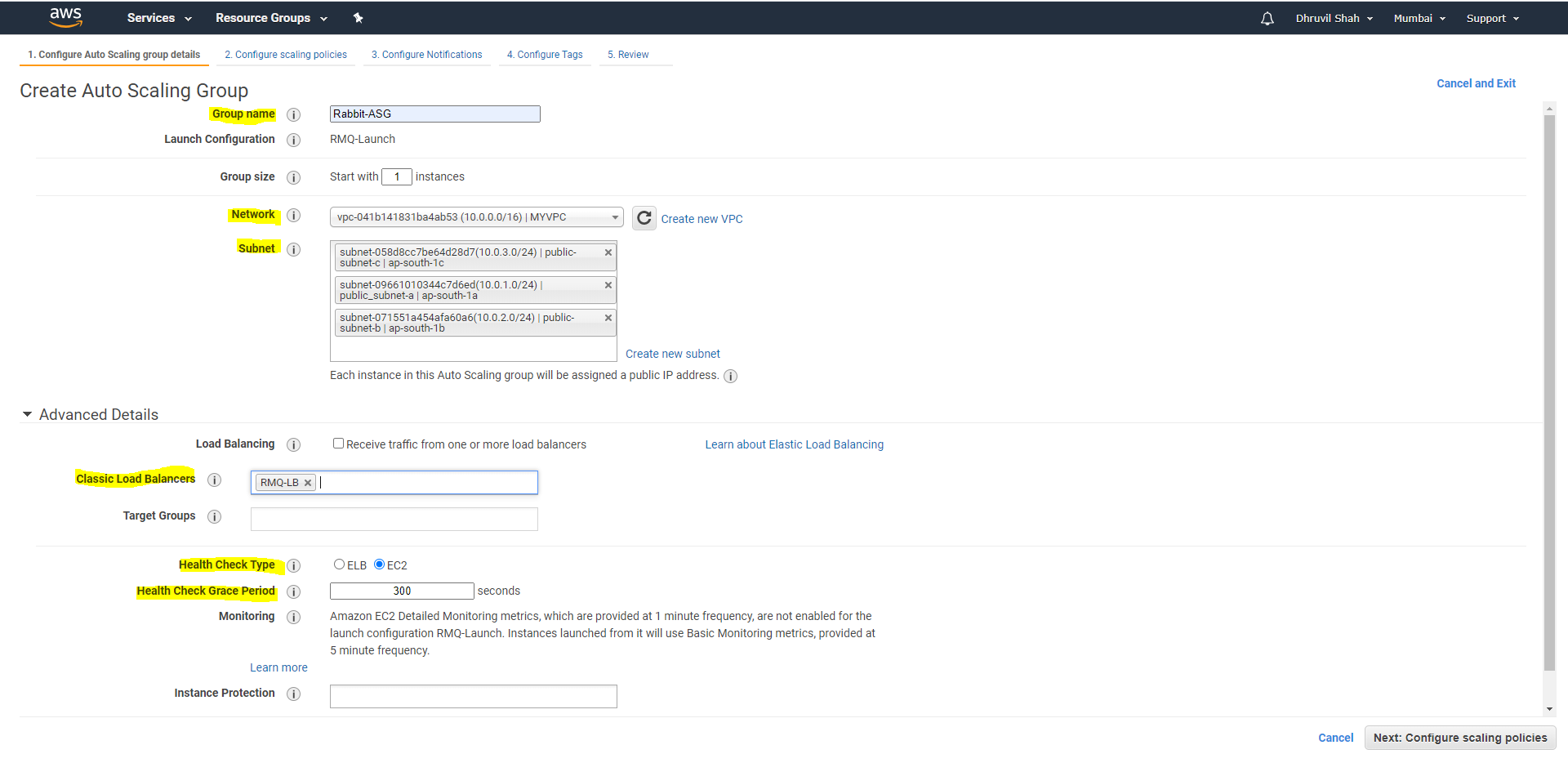
* Create the Auto scaling group.



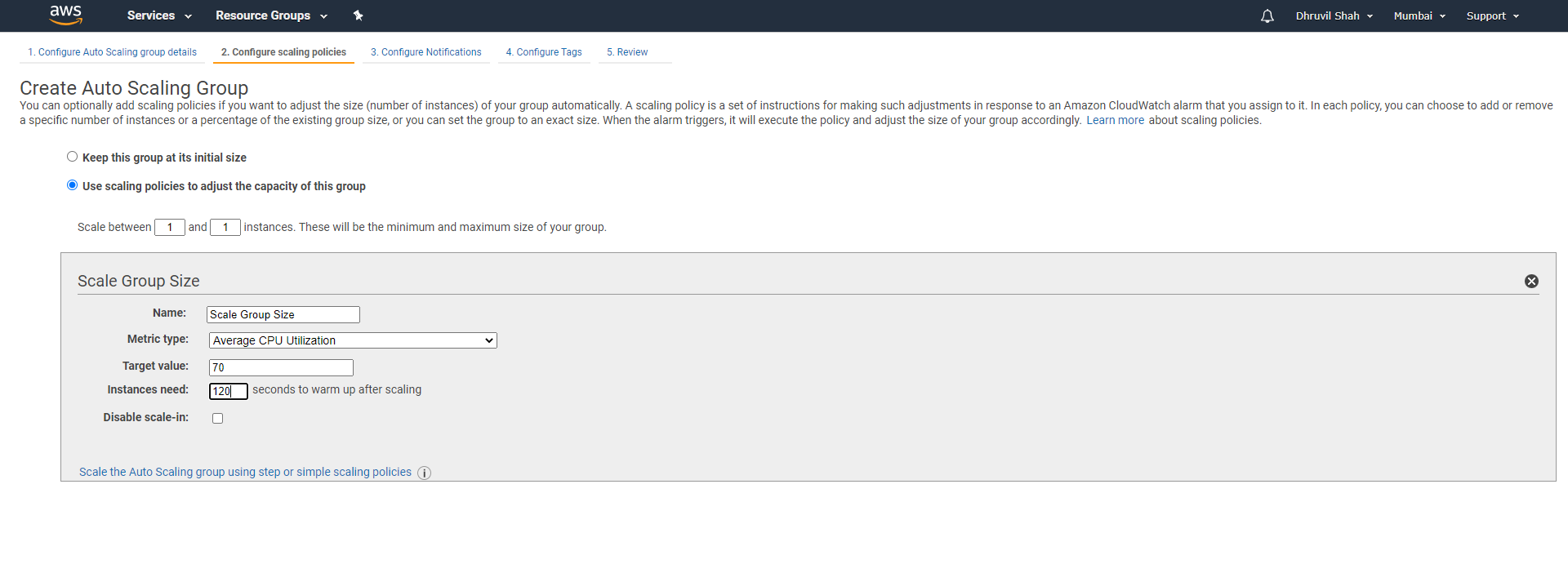
* Select the RMQ-Launch and click on next button.



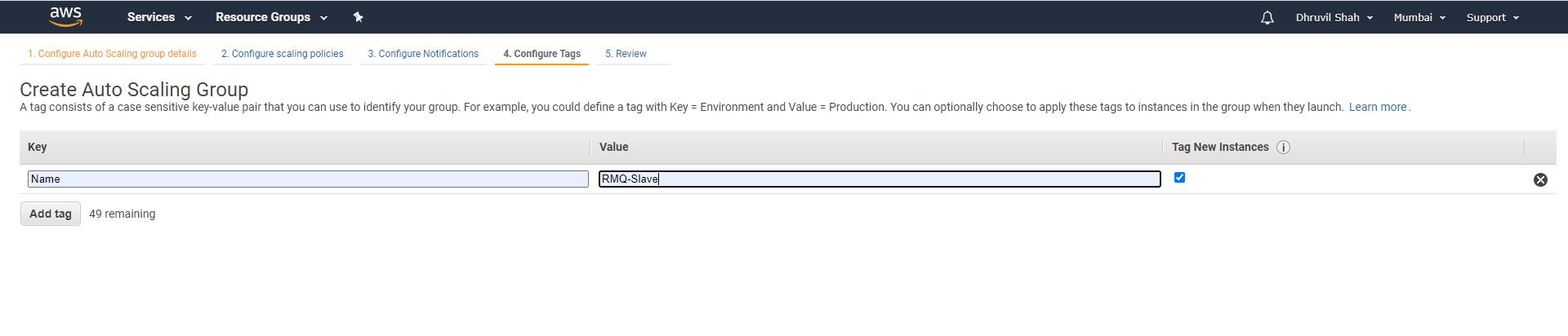
* Create the lunch Configuration and the select the network and subnet, health check.



* Create the UP-DOWN policy for the auto scaling .



* Create the tag for the instance.



* Review the Auto Scaling.

