D: 19/06/2020

3500 PUT/Sec (Uploading/overwriting)

5500 Get/Sec (Downloading)

EC2 : Elastic COmpute Cloud

--> CPU, Memory, Network, Disk AWS EC2

ec2 = virtual servers = Instance

server = Virtual machine = Azure VM = GCP engine = Instance

On-Demand ec2 instance : No stable workloads, unpredictable usages, temporary testing..

Pricing : /hr, /sec(Min of 60 Sec)

Reserved Instances : predictable worklods, stable configuration, longterm usages..

Pricing : 1 Yr / 3 Yrs..

Standard RI : We cannot change the configuration..

Convertable RI : We can modify the instance configuration..

Scheduled RI : repeated usages, persistant requests..

Pricing : Full Upfront : Pay 100% and get your instance..

Partial Upfront : pay 30-50% now, pay hourly basis with reduced price..

No Upfront : Pay everything monthly basis..

Spot Instances : If we have flexible start/stop durations..

We need to bid against the AWS pricing, When aws instance price is reduced or equal to customer quoted price, we will get this instance.

c4.4xlarge = 2$/hr.. 1$/hr, from date, to date..

--> Spot fleet request

--> Spot pricing history

EC2 is a region specific service.

--> Mumbai

--> NV

------------------------------------------------------------------------------------------

S3 performances:

We are creating S3 bucket, and we are upload/download. So basically it may have logs and thumbnails of the files as we use web servers, Ec2 instances.

We can perform 3500 PUT/Sec (Uploading/overwriting) ,5500 Get/Sec (Downloading)

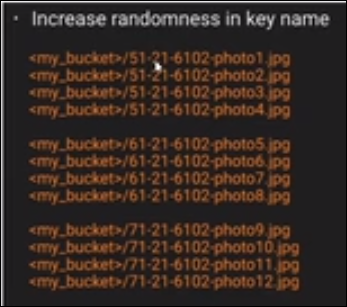
We may face performance issue; this can be happened when we don’t have proper folder structure, and naming pattern while upload objects into bucket.

Now I need to improve the performance.



Suppose any search algorithm is running then it takes time to get the data, as all the file names are almost similar except last suffix so this is not a recommended one,.so don’t place the object [files] into S3 bucket.

So one practice of naming is increase randomness in key name, as sown below.



Here just increased randomness, it is bit better compared to previous one naming type.

With in a bucket, we are creating the prefix, here prefix is nothing but adding a folder, ex:

Test/m1.com , here prefix is test and suffix is com.

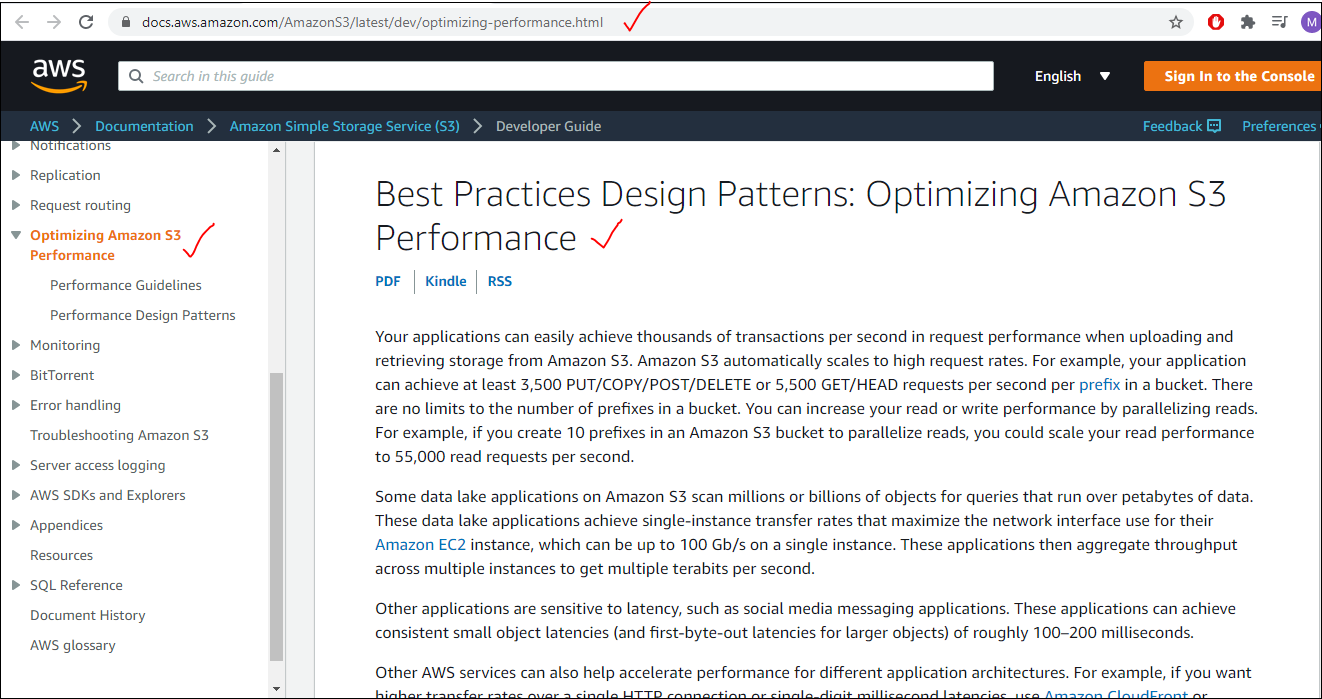
Suppose we have a folder which has 3.5k put, and 5.5k get, and we need to a performance as 10k then we can create folder in a folder so that each folder has 3.5k put, 5.5kget operations, so total operations will 3 times it will increase so it will have good performance.

Suppose we need the application to perform 55k read operations per second, so simple create 10 folders and segregates objects into them.



If we segregate in this manner we can get best performance. In organization level we will create the buckets like Prod : bucket1, Nprod:bucket2, UAT:bucket3,… however bucket naming will be unique.

<https://docs.aws.amazon.com/AmazonS3/latest/dev/optimizing-performance.html>



Prefix is not but folder, how many folders we create that much speed.

S3 will support unlimited storage, max limit for one single object is 5TB, but we can upload 10 – 5TB objects data or 100 -5 TB objects data in single bucket.

---------------------------------------------------------------------------------------------------------------------

**EC2 module**

It’s Stands for Elastic Compute Cloud.

What are computing resources means? CPU, Memeory,disk,Network…

So here in AWS Cloud we are getting all these computing resources.all these computing resources we can perform elastically. We can scale them up /down ,create them, /delete them. For ex: from very basic configuration to very high configuration with in minutes and seconds.

EC2 is a platform,where we can launch virtual servers.

In AWS we will call them as Instances.

Server = Instance = In AZure :AzureVM = In GCP :GCP Engine

An Instance is nothing but a server, we are not going to call them as server, we will call them as Instance.

As per our requirements we need to select the EC2. We have

**On-Demand ec2 instance**:

If we don’t know how many hours we are using the instance, and don’t know what type of configuration is required, testing the app for first time, and need to give a drive on the instance. In all these cases we can go for On-Demand ec2 instance.

Use case: No stable workloads, unpredictable usages, temporary testing..

Pricing: /hr, /sec(Min of 60 Sec), suppose we run a instance for 120sec we get bill for 120 sec.

Suppose we run an instance for 20 sec and deleted then we will get the bill for 60 sec. i.e., min billing is for 60 sec

Advantages : it is purely pay as go mechanism, based on the usage they will charge, suppose we run t he app on low configure and we are not stratified with that, we can immediately up scale for higher configuration, so billing will be according to the usage of the instance.

**Reserved Instances:**

So suppose we are happy to run the application in AWS EC2, and we decided to run it for another 5 or 20 years in AWS. But it is not recommended to go to hourly bases payment. so if we purchase a reserved instance, we will get a discount.

When we have predictable workloads, stable configuration, long term usages. We can prefer this Reserved Instance.

We have only two options of pricing for these Reserved Instance, Pricing : 1 Yr / 3 Yrs..

We have multiple types in this Reserved Instance.

1. Standard RI: We cannot change the configuration..
2. Convertible RI : We can modify the instance configuration.[ no downgrade the configuration, only upgrade the configuration is possible]
3. Scheduled RI: repeated usages, persistent requests.[ we have a requirement that where we need to run the server form Monday morning to Friday evening, and we need the server for 1 year, and at Friday evening it has to stop automatically, and again at Monday morning it has to start like a cycle for 1 year. so basically it is Term Time, which starts on Monday morning and ends on Friday evening. ]

So here we can opt for Standard/ Convertible RI, but we need to do the start and stop operations manually and that too they charge more. So mostly this RI is available only in North Virginia, and some two more places.

Pricing : we have three types.

1. Full Upfront: Pay 100% and get your instance.
2. Partial Upfront: pay 30-50% [ not fixed it might vary ] now, pay hourly basis with reduced price.
3. No Upfront: Pay everything monthly basis.

**Spot Instances:**

Here we need to bid against the AWS pricing, When AWS instance price is reduced or equal to customer quoted price, we will get this instance.

Suppose we have an instance like: c4.4xlarge it cost as 2$/hr and we are not ready to pay 2$/hr for this configuration, then we can make an offer that we will pay only 1$/hr from date to date. And when the price fluctuates, between the time duration from date to date, then that Instance price will become 1$,/80cents /60cents. Immediately Amazon will give that instance to us.

What happens if price increase, then Amazon will terminate that instance.

Use case:

If we have flexible start/stop durations. If instance stop also and not running/terminated/deleted also that is not going to affect our application or operations.

So we can do one thing suppose if we opt for max is 2$/hr, then what ever the price fluctuates between the max limit doesn’t stop our Instance/ terminate them.

--> Spot fleet request: the above one is the example for it.

--> Spot pricing history: we can observe the pricing form past days/months.

**🡪Under free Tier we get only ON demand EC2 Instance.**

🡪Most of the real time we prefer Reserved Instance. If it is POC then go for On demand RI.

Getting instance is different but he operations what we are performing is same for all Instances.

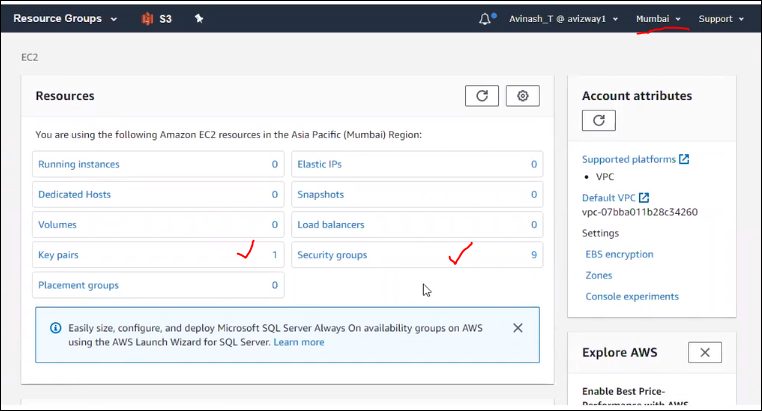
🡪we wont get schedule RI i.e., it is not available most of the time,

🡪Mostly we can see Standard RI. But most of the applications are upend running, suppose it is in prod env it is up end running, configuration also we will finalize when it is in staging area or testing env only .

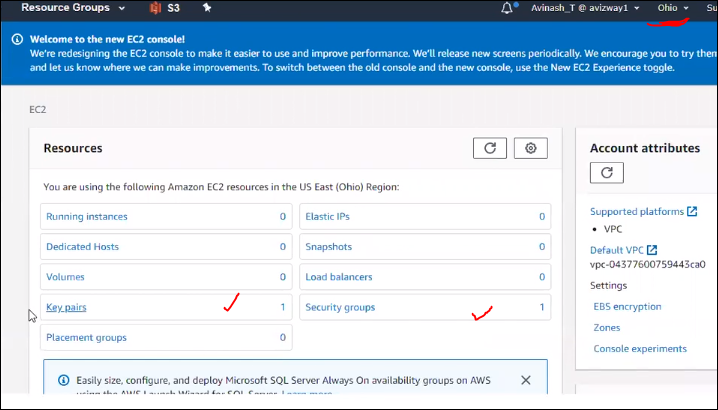
🡪In non prod, we can try all the options then we can finialize the one which suits for project.

**🡪EC2 is a region specific service.**

If we create an instance in one region it will not shown in another region.



Suppose if we switch the region to another one then see the difference

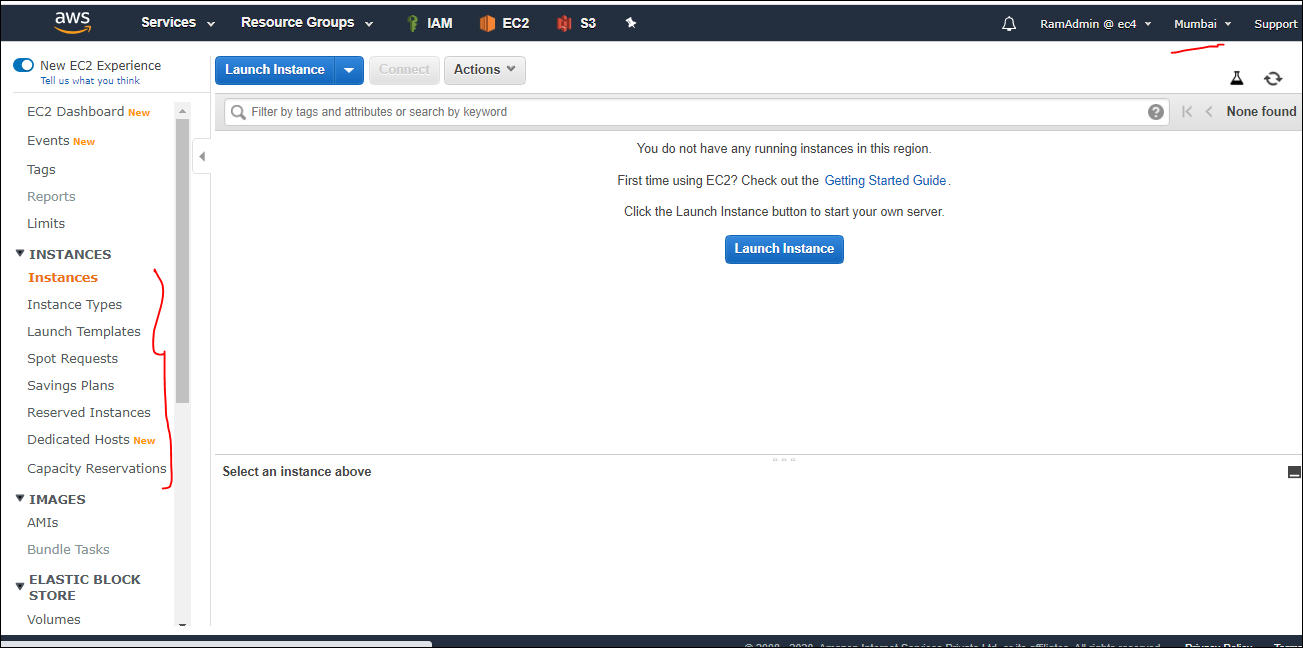


So suppose if we switch the region for some other purpose we might forgot to stop the running instance whichare running in another region.

Choose nearby region coz we can avoid latency.here we are going to perform Mumbai region.

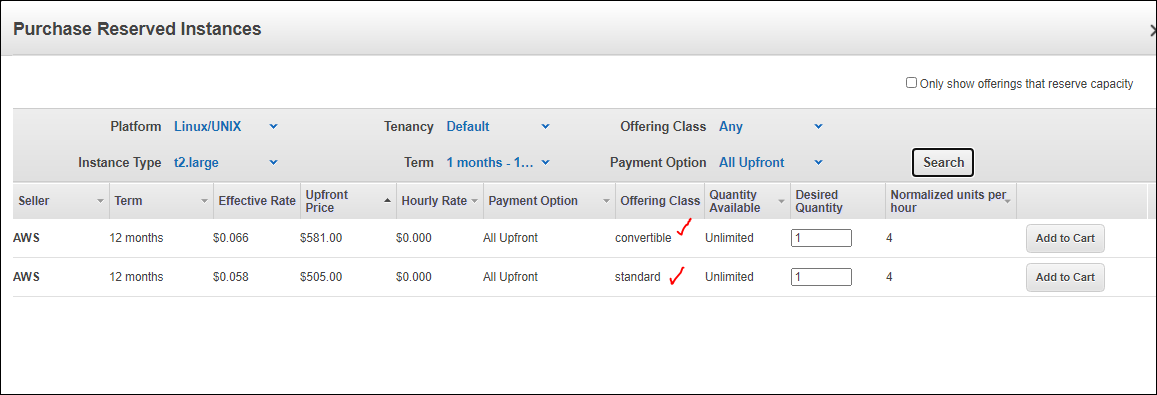
But some scenarios suppose we need to create the resources in multiple region.

Let us wlkthru all the above mentioned instances.

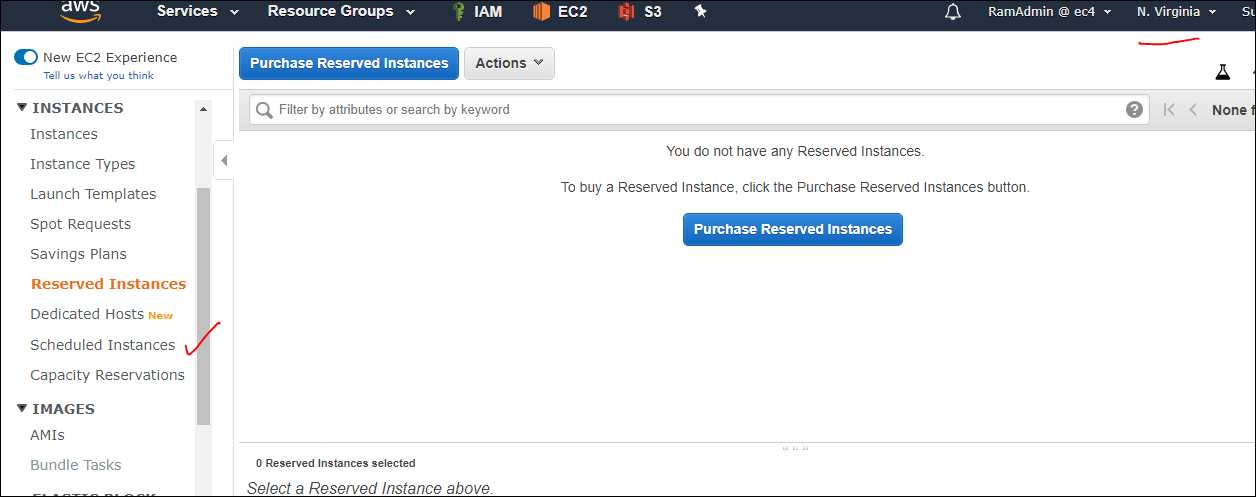


So if we follew the launch instance , that means we are launching On demand EC2 Instance.

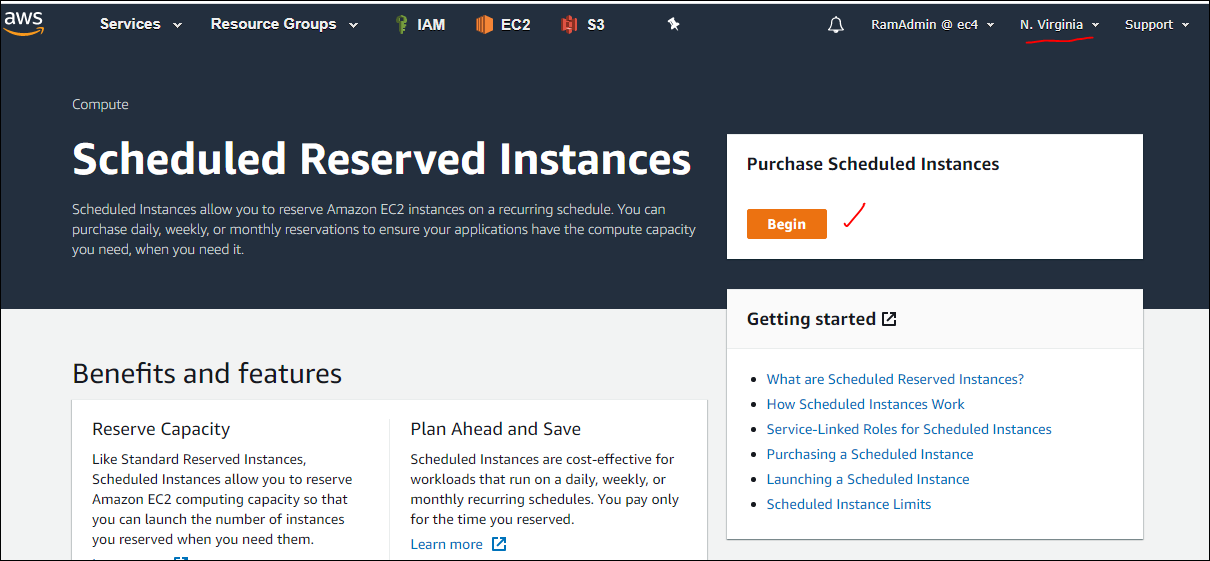
When we need Reserved then Under Instance 🡪Reserved instance. Click on Purchase Reserved Instance and select the required options.



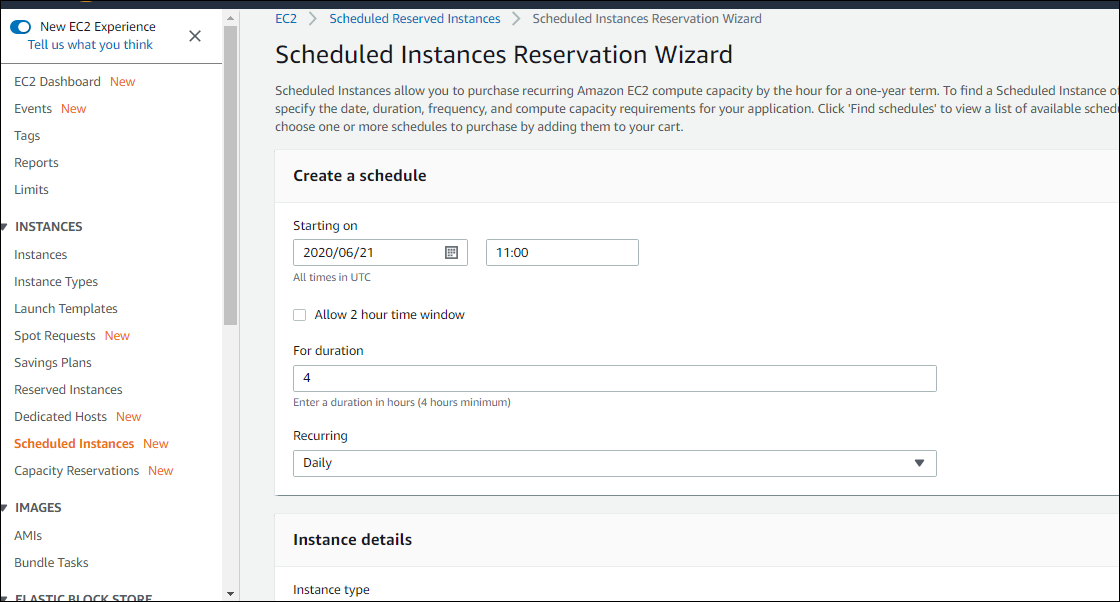
This is how we purchase the reserved Instance, if we need to see the schedule RI then we needs to switch the Region coz it is available only in selected regions.



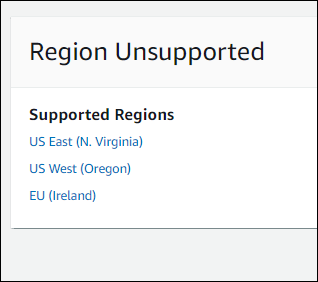
Click on Scheduled instances.



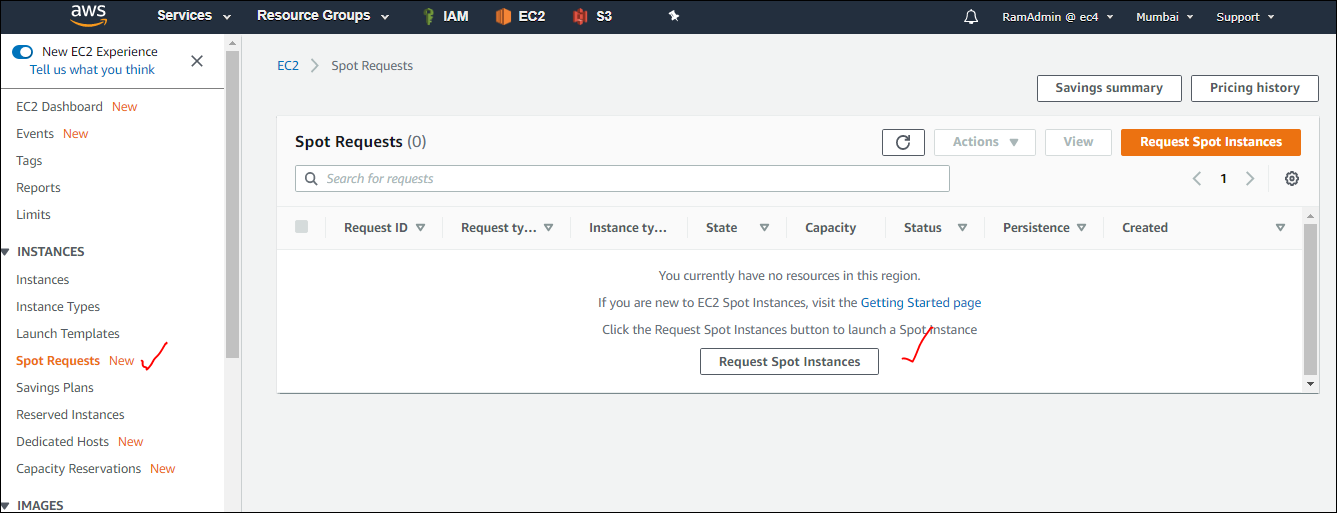
Based on the requirement we can select the options and schedule the resierved Instance.



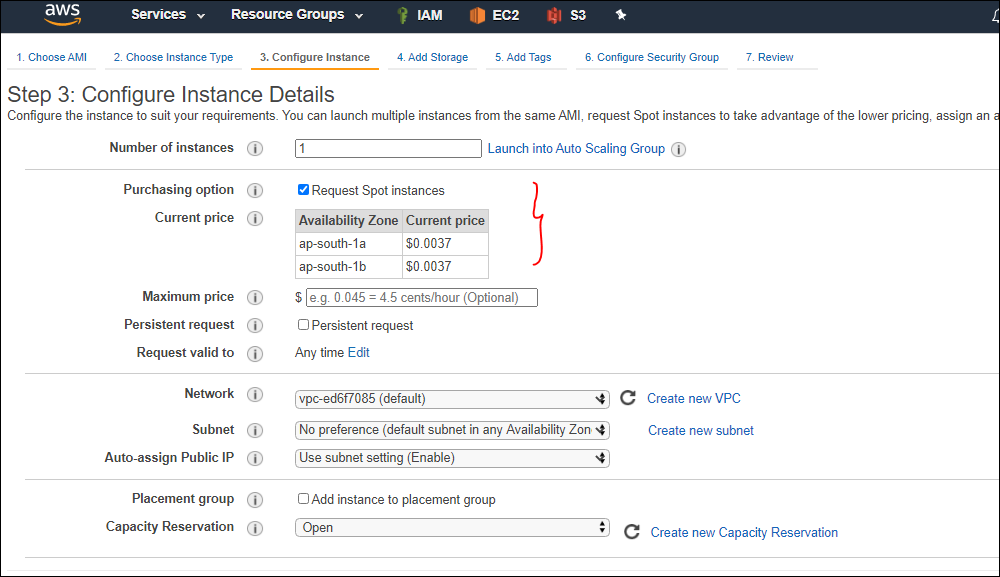
These are the regions supported by schedule RI as listed below.

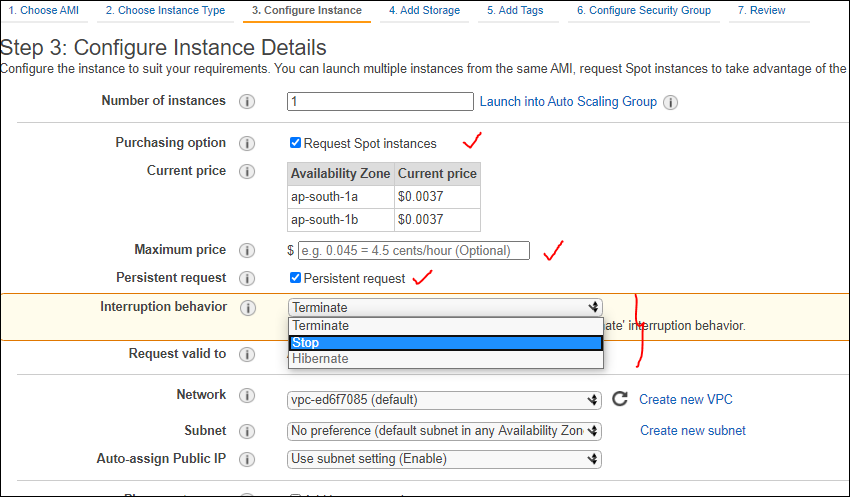


Coming to sport instance,

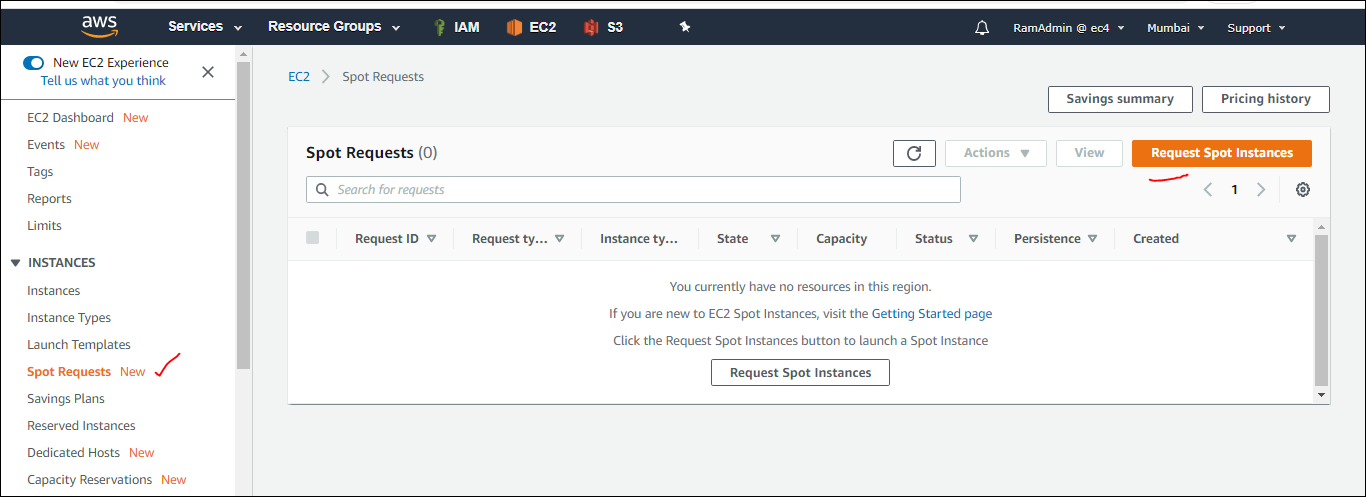


Even in purchasinf instance , we can have an option to select the Spot Instance.

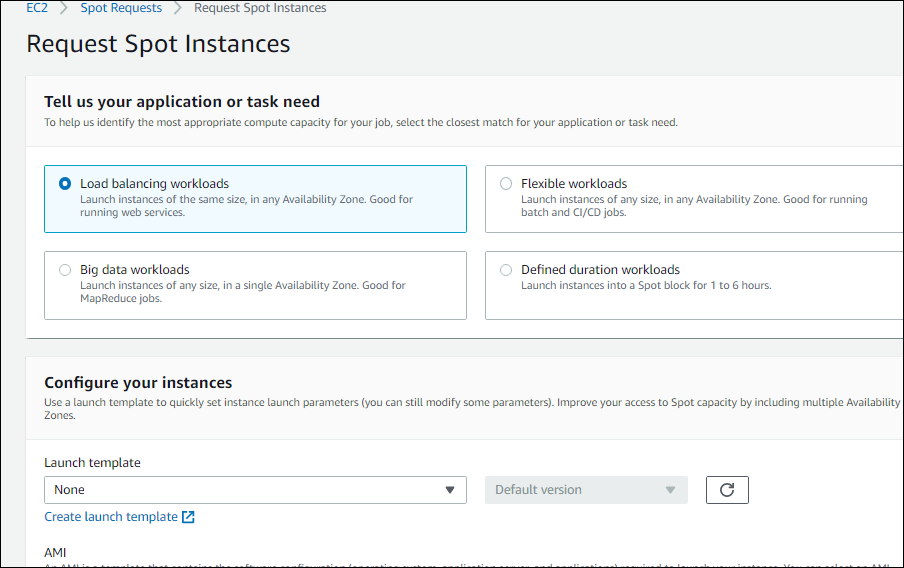




Instead of that if we request a Fleet request all that then we can select it go to **Spot Request**



Click on request spot instance and provide the required values.



Even we can have a look on the pricing History

