Ever wonder why the apps of big companies like google, amazon e-commerce, meta and many more never or minimally face the down-time instead of having million and billions of concurrent users. If yes stay along with me to get it answered. We all know that the apps are launched on ever running servers in the data centers but we also know that these servers have limitation that at a specific time they can handle the fixed amount of traffic. If the traffic gets more than the server can handle it will crash. To handle this issue companies uses the concept of scaling (horizontal and vertical) and also uses the technology named Kubernetes where the pods get re-launched and the traffic is load balanced across various pods but that will be the discussion for later days. Today our focus area is scaling. Let’s discuss that in brief and also we will be looking at how to practically implement that with the help of AWS EC2 service automated with the help of python script. The code will be shared as I will drop the link of Git-hub repository. Now let’s discuss the concept of scaling.

**Scaling** refers to the process of increasing or decreasing the capacity of a system, resource, or application to handle varying levels of demand or workload. It’s a key concept in technology, particularly in **cloud computing**, **software development**, and **business growth strategies**.

In computing, scaling is used to ensure that applications and systems perform well as the number of users or data grows.

* **Vertical Scaling (Scale-Up):** Adding more resources to a single machine (e.g., increasing CPU, RAM, or storage).  
  Example: Upgrading a server to a higher configuration.
* **Horizontal Scaling (Scale-Out):** Adding more machines or instances to distribute the load.  
  Example: Adding multiple servers to a load balancer.
* **Auto-Scaling:** Automatically adjusting resources based on demand (common in cloud environments like AWS, Azure, or Google Cloud).

Today we will see the code for horizontal scaling.

Here is the flow for the process.

Pre-requisites: AWS account to be authenticated in local system, boto3 python library should be downloaded using “pip install boto3”. And the definitely python should be on the system you are using.

Process: import the boto3 module.

Create an object of boto3.client() class by passing the service which we want to use in our case ec2 and same way create an instance of boto3.resource() class with the parameter as the resource which we want to use.

Will create instances of ec2 by providing the following  
ImageId, Imagetype, MaxCout and minCount

Same way we can delete the instances which are not required by passing the instanceId

See the code below to get better understanding.

url: