

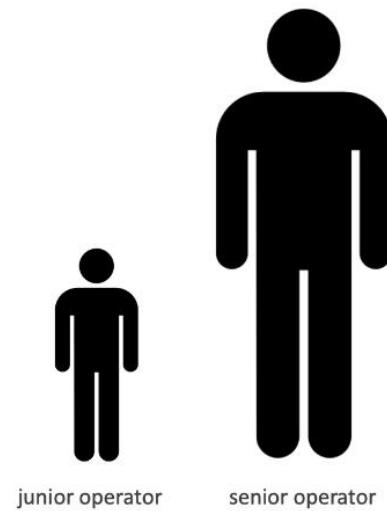
Scalability & High Availability

- Scalability means that an application / system can handle greater loads by adapting.
- There are two kinds of scalability:
 - Vertical Scalability
 - Horizontal Scalability (= elasticity)
- Scalability is linked but different to High Availability
- Let's deep dive into the distinction, using a call center as an example



Vertical Scalability

- Vertical Scalability means increasing the size of the instance
- For example, your application runs on a t2.micro
- Scaling that application vertically means running it on a t2.large
- Vertical scalability is very common for non distributed systems, such as a database.
- There's usually a limit to how much you can vertically scale (hardware limit)



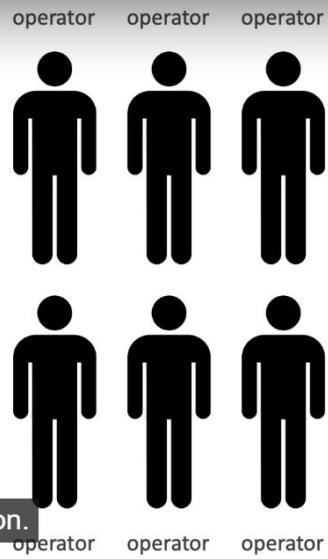
Okay, next is horizontal scalability.

Junior upgraded to senior, senior can handle more calls.

If want more performance to DB, just increase the size of DB

Horizontal Scalability

- Horizontal Scalability means increasing the number of instances / systems for your application
- Horizontal scaling implies distributed systems.
- This is very common for web applications / modern applications
- It's easy to horizontally scale thanks the cloud offerings such as Amazon EC2
and we'll see this in the section.



1 / 6:23 © Jane Maarek



If want to handle more calls, add more operators

High Availability

- High Availability usually goes hand in hand with horizontal scaling
- High availability means running your application / system in at least 2 Availability Zones
- The goal of high availability is to survive a data center loss (disaster)



And in AWS, it could be an earthquake,

1 / 6:23 © Jane Maarek



If new York fail, still take calls in san Fransisco

High Availability & Scalability For EC2

- Vertical Scaling: Increase instance size (= scale up / down)
 - From: t2.nano - 0.5G of RAM, 1 vCPU
 - To: u-12tb1.metal – 12.3 TB of RAM, 448 vCPUs
- Horizontal Scaling: Increase number of instances (= scale out / in)
 - Auto Scaling Group
 - Load Balancer
- High Availability: Run instances for the same application across multi AZ
 - Auto Scaling Group multi AZ
 - Load Balancer multi AZ

And a load balancer in multi AZ.

Auto Scaling is a cloud computing feature that **automatically adjusts the number of running instances or resources** in response to real-time demand — **scaling up** when demand increases and **scaling down** when demand drops.

Scalability vs Elasticity (vs Agility)

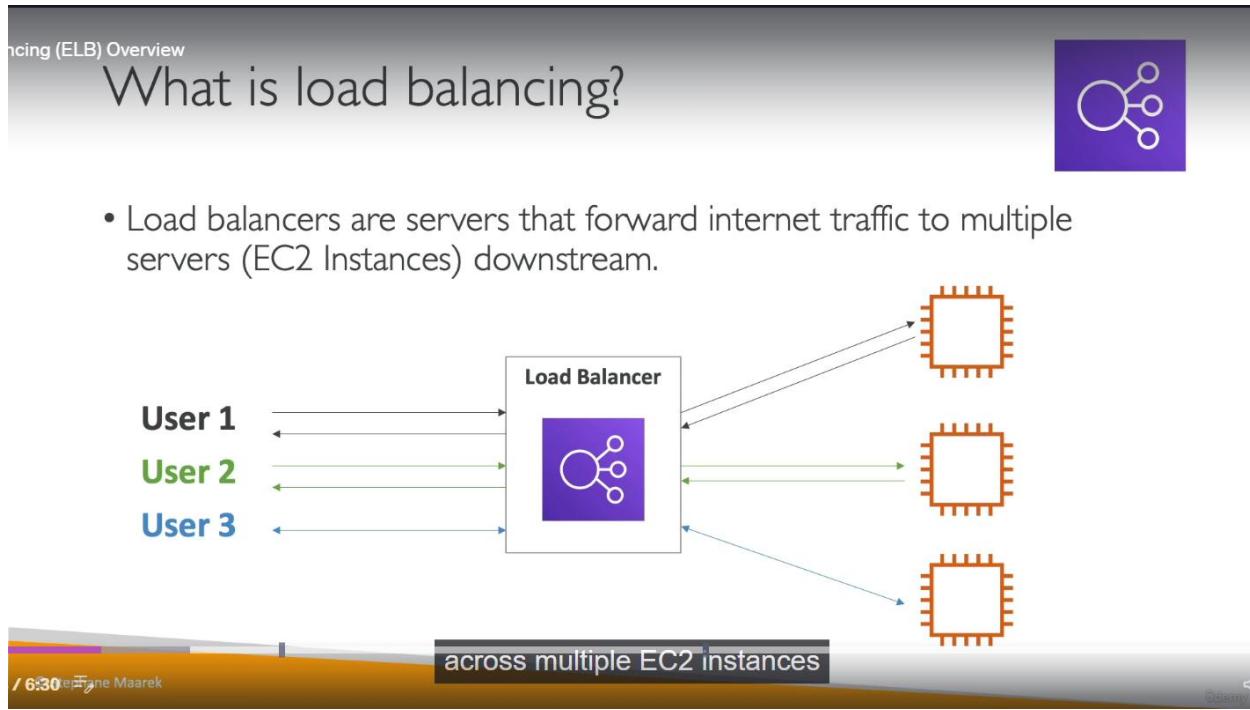
- **Scalability:** ability to accommodate a larger load by making the hardware stronger (scale up), or by adding nodes (scale out)
- **Elasticity:** once a system is scalable, elasticity means that there will be some “auto-scaling” so that the system can scale based on the load. This is “cloud-friendly”: pay-per-use, match demand, optimize costs
- **Agility:** (not related to scalability - distractor) new IT resources are only a click away, which means that you reduce the time to make those resources available to your developers from weeks to just minutes.

to make these resources available to your developers

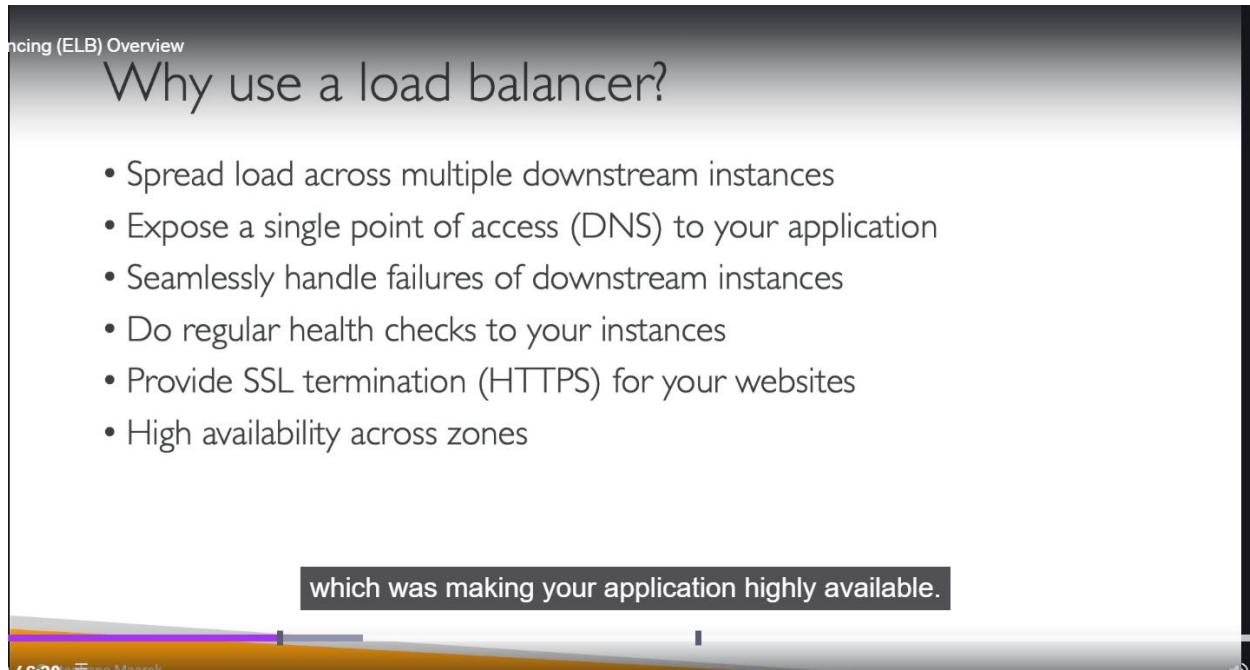
ELASTIC LOAD BALANCING(ELB)

Allows to be elastic

also called backend EC2 instance



more users, more it will balance the load,..allow us to scale better our backend

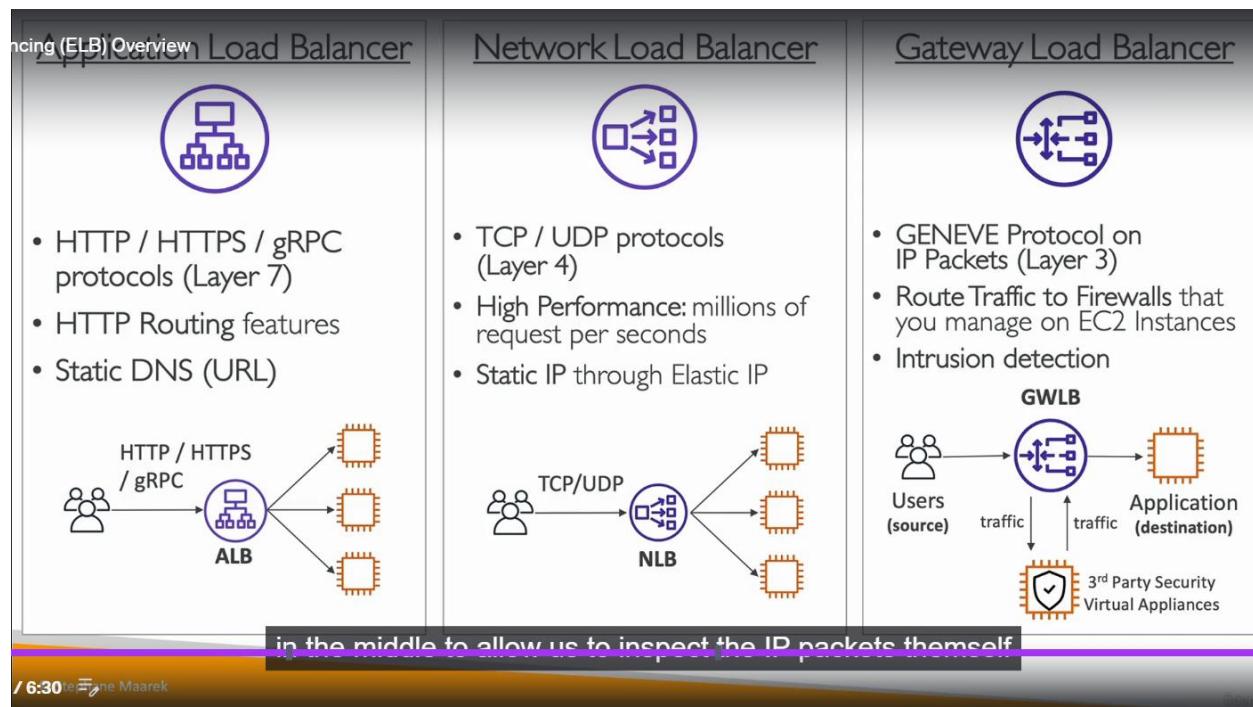


If one instance fails, don't send to that instance, so hiding our failure

Why use an Elastic Load Balancer?

- An ELB (Elastic Load Balancer) is a **managed load balancer**
 - AWS guarantees that it will be working
 - AWS takes care of upgrades, maintenance, high availability
 - AWS provides only a few configuration knobs
- It costs less to setup your own load balancer but it will be a lot more effort on your end (maintenance, integrations)
- 4 kinds of load balancers offered by AWS:
 - Application Load Balancer (HTTP / HTTPS only) – Layer 7
 - Network Load Balancer (ultra-high performance, allows for TCP) – Layer 4
 - Gateway Load Balancer – Layer 3
 - Classic Load Balancer (**and Layer 7 type of load balancer of older generation.**)

Only thing we have to do in configure



GLB inspects IP packets and to perform firewall features or intrusion detection or deep packet inspection

🚀 What is an Auto Scaling Group (ASG)?

An **Auto Scaling Group** (used in AWS and other clouds) is a **collection of virtual machines (instances)** that are managed together and **automatically scaled** based on demand.

 **Key Features:**

Feature	Description
Auto-scaling	Adds/removes instances automatically based on CPU, memory, or custom metrics
Minimum and Maximum Size	You define how few and how many instances the group can have
Health Checks	Unhealthy instances are replaced automatically
Integration with Load Balancer	Works with load balancer to route traffic efficiently

 **Example Use Case:**

You have a website. In the evening, user traffic spikes. The **ASG launches 3 more servers**. Later at night, traffic drops — **ASG terminates those extra servers** to save cost.

 **What is a Load Balancer?**

A **Load Balancer** is a service that **distributes incoming traffic across multiple servers** to ensure:

- No single server is overloaded
 - High availability and fault tolerance
 - Faster response times
-

 **Types of Load Balancers:**

Type	Description
Application Load Balancer (ALB)	Routes based on HTTP/HTTPS content (e.g., URL paths, host headers)
Network Load Balancer (NLB)	Routes based on IP and TCP/UDP ports — lower level, high performance
Classic Load Balancer (CLB)	Older, basic version for simple traffic distribution

❖ How They Work Together:

1. **Users** access your web app.
2. Traffic hits the **Load Balancer**.
3. Load Balancer forwards it to one of the **instances in the Auto Scaling Group**.
4. If traffic increases, **ASG adds instances** and Load Balancer includes them automatically.
5. If traffic drops, **ASG removes instances** and Load Balancer stops sending traffic to them.

ASG

ups (ASG) Overview

What's an Auto Scaling Group?



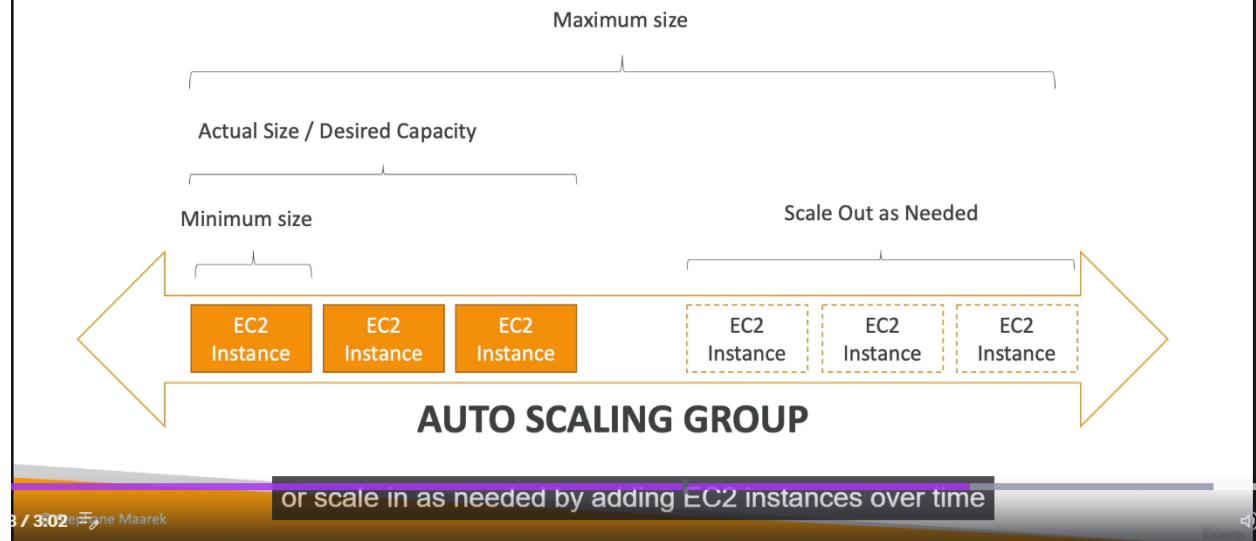
- In real-life, the load on your websites and application can change
- In the cloud, you can create and get rid of servers very quickly
- The goal of an Auto Scaling Group (ASG) is to:
 - Scale out (add EC2 instances) to match an increased load
 - Scale in (remove EC2 instances) to match a decreased load
 - Ensure we have a minimum and a maximum number of machines running
 - Automatically register new instances to a load balancer
 - Replace unhealthy instances
- Cost Savings: only run at an optimal capacity (principle of the cloud)

because we are only running all the time

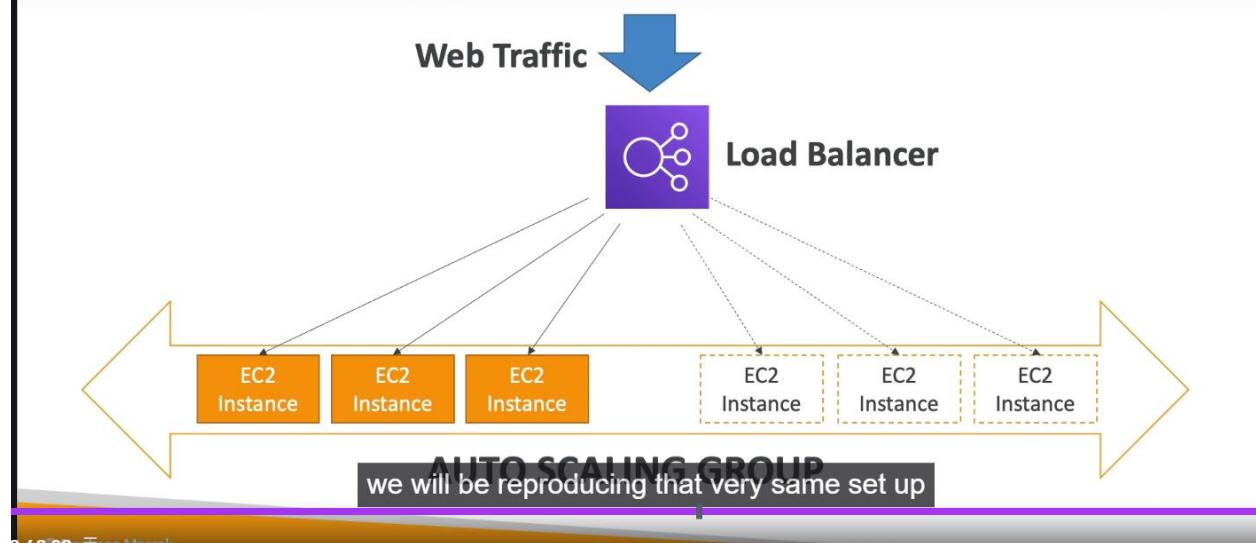
2 / 3:02 e Bone Maarek

more shopping at day and not at night

Auto Scaling Group in AWS



Auto Scaling Group in AWS With Load Balancer



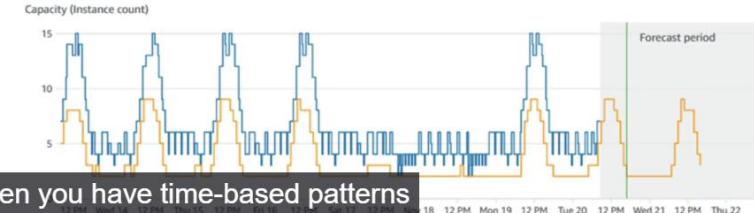
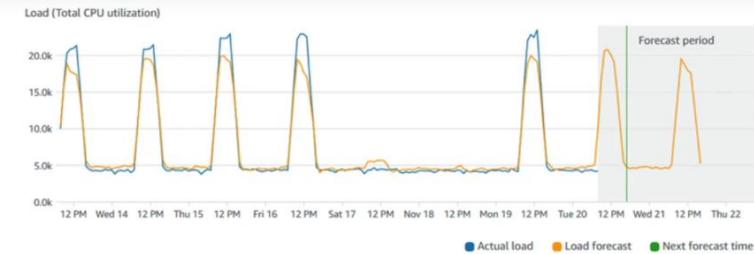
Auto Scaling Groups – Scaling Strategies

- Manual Scaling: Update the size of an ASG manually
 - Dynamic Scaling: Respond to changing demand
 - Simple / Step Scaling
 - When a CloudWatch alarm is triggered (example CPU > 70%), then add 2 units
 - When a CloudWatch alarm is triggered (example CPU < 30%), then remove 1
 - Target Tracking Scaling
 - Example: I want the average ASG CPU to stay at around 40%
 - Scheduled Scaling
 - Anticipate a scaling based on known usage patterns
 - Example: increase the min capacity to 10 at 5 pm on Fridays
- People are going to do sports betting maybe who knows,**

Changing capacity from 1 to 2 or etc

Auto Scaling Groups – Scaling Strategies

- Predictive Scaling
 - Uses Machine Learning to predict future traffic ahead of time
 - Automatically provisions the right number of EC2 instances in advance
- Useful when your load has predictable time-based patterns



when you have time-based patterns

Elasticity is scaling up and down

ELB & ASG – Summary

- High Availability vs Scalability (vertical and horizontal) vs Elasticity vs Agility in the Cloud
- Elastic Load Balancers (ELB)
 - Distribute traffic across backend EC2 instances, can be Multi-AZ
 - Supports health checks
 - 4 types: Classic (old), Application (HTTP – L7), Network (TCP – L4), Gateway (L3)
- Auto Scaling Groups (ASG)
 - Implement Elasticity for your application, across multiple AZ
 - Scale EC2 instances based on the demand on your system, replace unhealthy
 - Integrated with the ELB

elasticity, and agility in the Cloud.