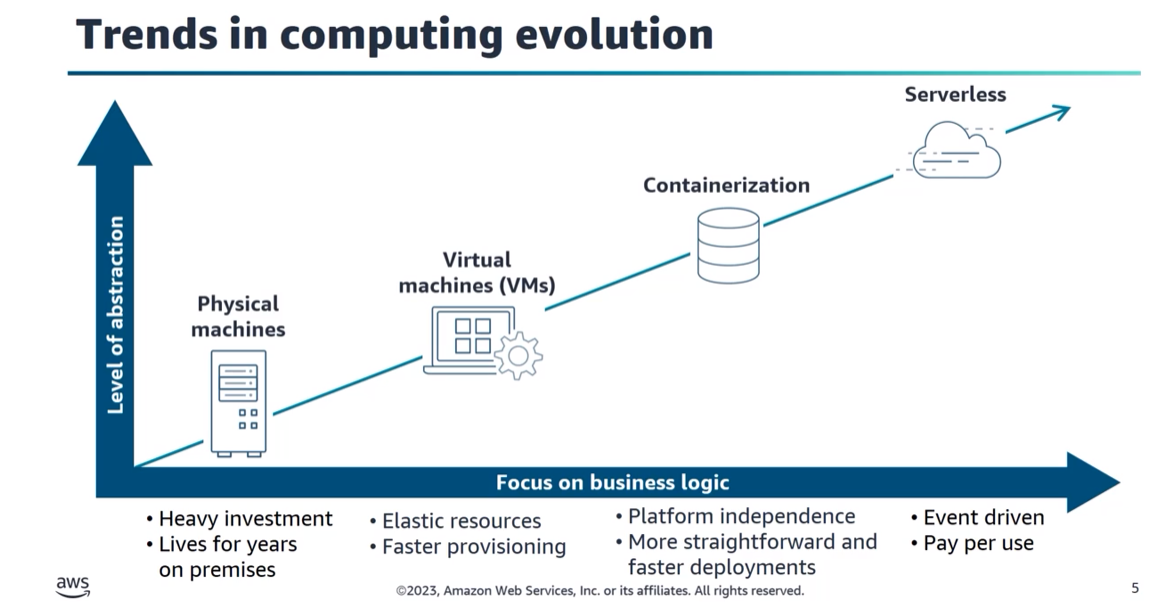
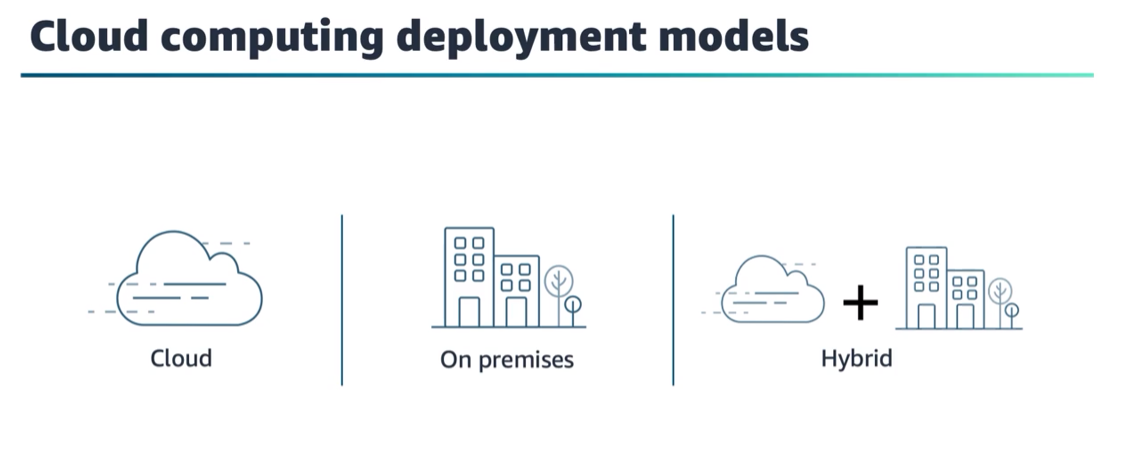
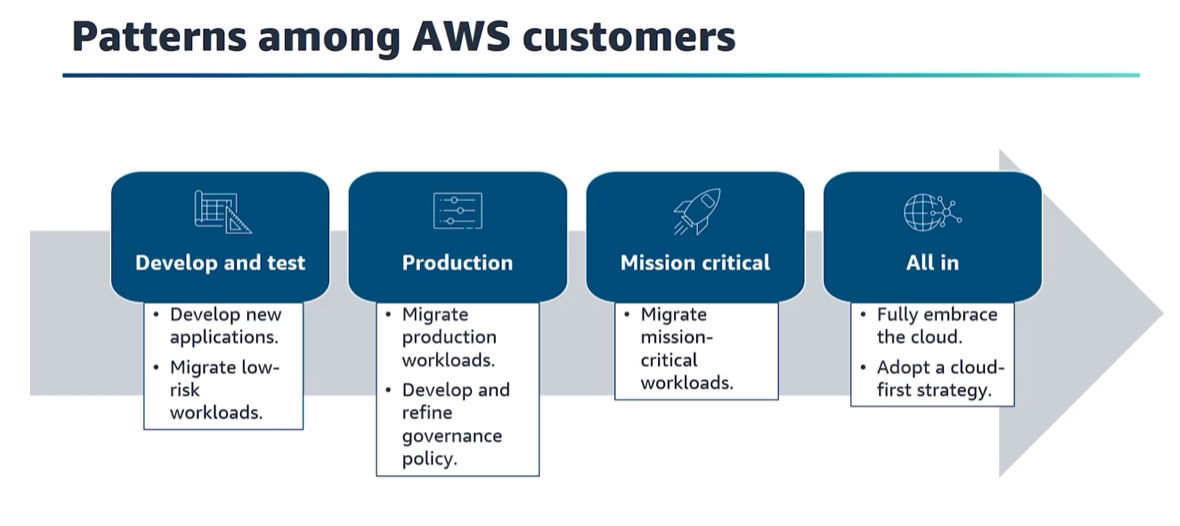
# **AWS Certified Cloud Practitioner**

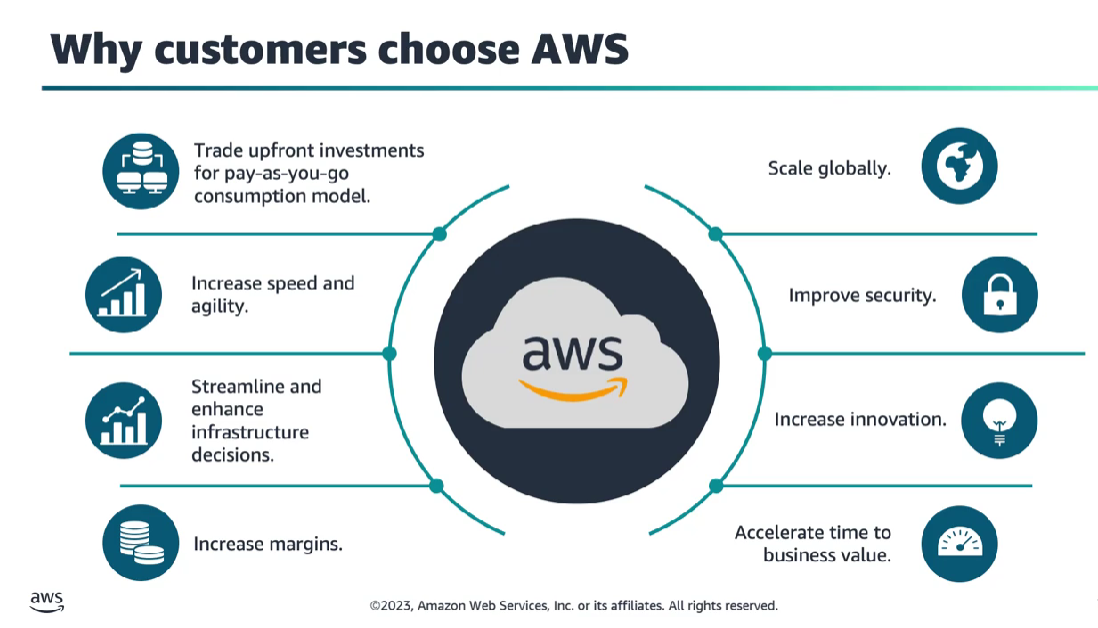
## **Cloud Basics and AWS**

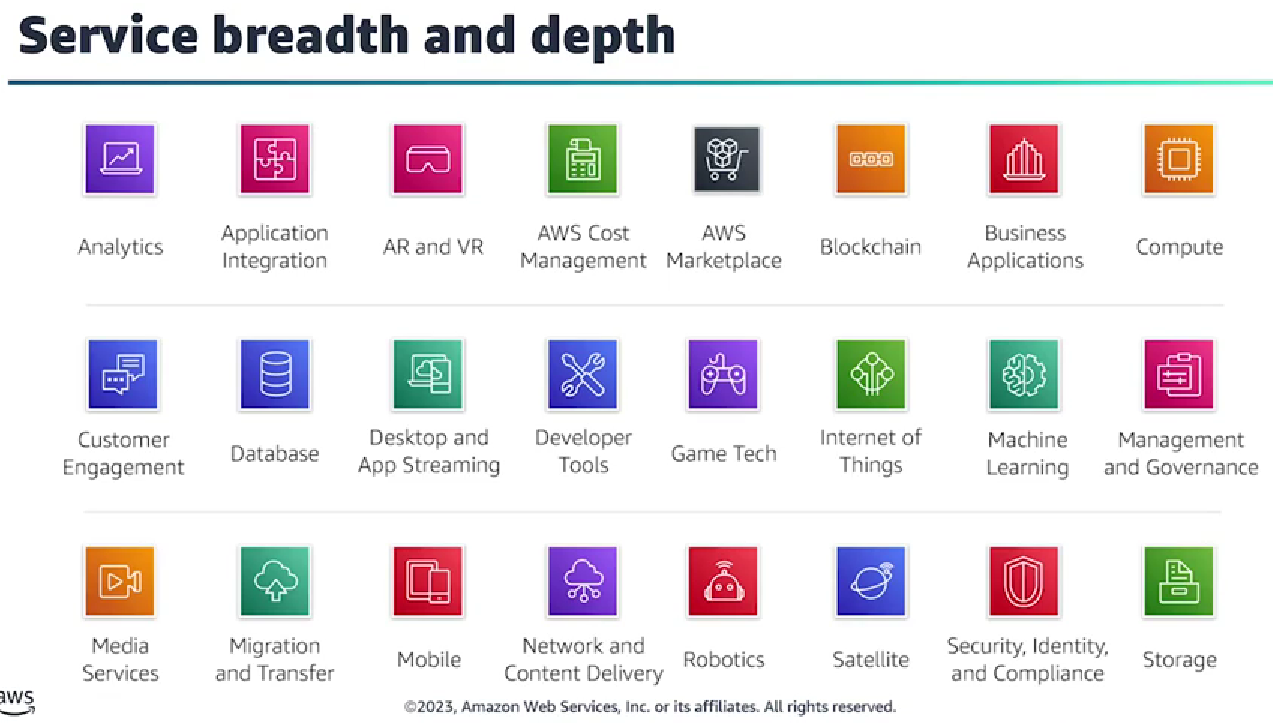
* **What is Cloud Computing:** It is the on-demand delivery of IT resources over the internet with pay-as-you-go pricing. The cloud includes services beyond the infrastructure.

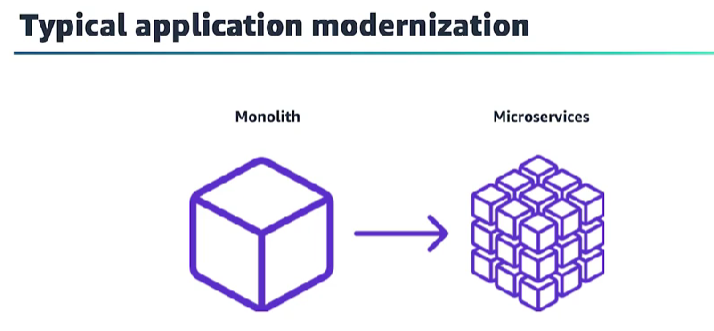
****

****

****

****

****

****

* Customers are going through a major shift—an architecture evolution. They are moving from one, monolithic application to smaller and independent microservices. **Microservices** are minimal-function services that are deployed separately but can interact together to achieve a broader use case. These applications then become more straightforward to build and maintain because they are smaller and more manageable.
* **Some limitations to monolith applications include the following:**
* **Are hard to scale**
* **Cannot handle component failures**
* **Have a slow deployment process**
* **Have limited options**

AWS offers a complete platform for microservices. It provides the building blocks to support the development of microservices with compute, storage, database, networking, messaging, logging and monitoring, and DevOps services.

AWS the Most Complete Platform for Microservices

1. **Compute:**

### **Containers**

Amazon Elastic Container Service (EC2)

### **Serverless**

AWS Lambda

1. **Storage & Databases:**

### **Object Storage**

Amazon S3

### **Caching**

Amazon ElastiCache

### **Relational Databases**

Amazon RDS

### **NoSQL Databases**

Amazon DynamoDB

**And many more….**

* **IaaS (Infrastructure as a Service): Amazon EC2 (Elastic Compute Cloud)** provides scalable virtual servers in the cloud, allowing users to run applications and manage their own operating systems.
* **PaaS (Platform as a Service): AWS Elastic Beanstalk** simplifies the deployment and management of applications by providing a fully managed platform, including runtime environments and infrastructure configuration.
* **SaaS (Software as a Service): Amazon WorkMail** is a SaaS offering that provides email and calendaring functionality, eliminating the need for organizations to manage their email infrastructure.

| **Category** | **IaaS** | **PaaS** | **SaaS** |
| --- | --- | --- | --- |
| **Compute** | Amazon EC2 | Heroku | Salesforce |
| **Storage** | Amazon S3, Azure Blob Storage | Google Cloud Storage, Microsoft Azure Storage | Dropbox, Google Drive |
| **Database** | Amazon RDS, Azure SQL Database | Google Cloud SQL, Heroku Postgres | Salesforce CRM, Microsoft 365 |
| **Networking** | Amazon VPC | Google Cloud Networking, Azure Virtual Network | Cisco WebEx, Zoom |
| **Development Tools** | AWS CloudFormation, Google Deployment Manager | Heroku, Microsoft Azure DevOps | GitHub, GitLab |
| **Middleware** | Amazon MQ, Azure Service Bus | Google Cloud Pub/Sub, Azure Event Grid | Salesforce App Cloud, Microsoft Azure App Service |
| **Analytics** | Amazon Redshift, Google BigQuery | Microsoft Azure Synapse Analytics | Google Analytics, Salesforce Analytics Cloud |
| **Machine Learning** | Amazon SageMaker, Google AI Platform | Microsoft Azure Machine Learning | Salesforce Einstein Analytics |

**AWS cost optimization:**

* Choose the right pricing model (Pay-as-you-go)
* Match capacity with demand
* Implement processes to identify resource waste.

**Most AWS customers run hybrid infrastructures, meaning that some of it is in more traditional on-premises data centers and other parts are in the cloud.**

**Moving to AWS does not mean an all-or-nothing move and getting rid of your customer’s present investment. It is about optimizing and streamlining their infrastructure, meaning they move to the cloud only what makes sense.**

**Switching Costs and Lock-In:**

It’s no surprise that organizations are worried about becoming locked in to their cloud provider. After all, the history of IT is full of examples of vendors taking advantage of high switching costs to impose restrictive licensing terms and to increase prices. But I think that the cloud is different—and in fact, is making it harder and harder for software, hardware, and IT service vendors to take advantage of the leverage that they have had in the past.

As soon as you commit yourself to a platform or a vendor you will have switching costs if you later decide to change. but switching costs can be high or low. Through good design and some advance thought, you can reduce your costs of switching (from traditional software or from a cloud provider).

## **AWS Cloud Practitioner Essentials**

**What is a client-server model?**

* In computing, a**client** can be a **web browser** or **desktop application** that a person interacts with to make requests to computer servers.
* A **server** can be **services**, such as Amazon **Elastic Compute Cloud** (Amazon **EC2**) – a type of virtual server.

# ---------------------------------------------------------

**Amazon Elastic Compute Cloud (Amazon EC2)**

[Amazon Elastic Compute Cloud (Amazon EC2)(opens in a new tab)](https://aws.amazon.com/ec2/) provides secure, resizable compute capacity in the cloud as Amazon EC2 instances.

**Common EC2 instance types:**

1. **General Purpose Instances:**
   * t4g, t3, t3a, t2: These instances are well-suited for a diverse set of applications and workloads, including web servers, development environments, and small to medium-sized databases.
2. **Compute Optimized Instances:**
   * c7g, c6g, c5, c5a, c4: These instances are optimized for compute-intensive workloads, such as high-performance front-end fleets, web servers, batch processing, and scientific modeling.
3. **Memory Optimized Instances:**
   * u4sg, u4gd, u-6tb1.metal, u-9tb1.metal, u-12tb1.metal, r7g, r6g, r5, r5a, x1e, u-9tb1.metal, z1d: These instances are designed for memory-intensive applications, such as in-memory databases, real-time big data analytics, and high-performance computing (HPC) workloads**.**
4. **Storage Optimized Instances:**
   * i3, i3en, i3.metal, d2, h1: These instances are optimized for workloads that require high, sequential read and write access to very large data sets, such as data warehousing, distributed file systems, and big data processing.
5. **Accelerated Computing Instances:**
   * p4, p3, p3dn, inf1, f1: These instances use specialized hardware accelerators, such as GPUs or FPGAs, and are suitable for compute-intensive workloads like machine learning, high-performance computing, and video transcoding.

**Common pricing models for EC2 instances:**

1. **On-Demand Instances:**
   * This is a pay-as-you-go model where you pay for compute capacity by the hour or by the second (with a one-hour minimum). It is suitable for applications with short-term, irregular workloads, or where the user is uncertain about their usage patterns.
2. **Reserved Instances:**
   * With Reserved Instances (RIs), you commit to a one- or three-year term, and in return, you receive a significant discount compared to On-Demand pricing. RIs are a good choice for steady-state or predictable workloads.
3. **Spot Instances:**
   * Spot Instances allow you to bid for unused EC2 capacity at potentially lower costs than On-Demand instances. However, they can be terminated by AWS if the capacity is needed by On-Demand or Reserved Instances. Spot Instances are suitable for fault-tolerant and flexible applications.
4. **Dedicated Hosts:**
   * Dedicated Hosts enable you to have dedicated physical servers for your use. This can be beneficial for compliance requirements or to meet licensing agreements. Pricing for Dedicated Hosts can be on-demand or reserved.
5. **Savings Plans:**
   * Savings Plans provide significant savings over On-Demand pricing (up to 72%) in exchange for a commitment to a consistent amount of compute usage (measured in $/hr) for a 1 or 3-year term. They offer flexibility across instance families, sizes, and regions.
6. **Instance Savings Plans:**
   * Instance Savings Plans provide savings similar to Reserved Instances, but offer more flexibility in terms of instance family, size, and region. They provide a discount in exchange for a commitment to a consistent amount of compute usage for a 1 or 3-year term.

**Amazon EC2 Auto Scaling**

Amazon EC2 Auto Scaling enables you to automatically add or remove Amazon EC2 instances in response to changing application demand.

* *Dynamic scaling* responds to changing demand.
* *Predictive scaling*automatically schedules the right number of Amazon EC2 instances based on predicted demand.

**Example: Amazon EC2 Auto Scaling**

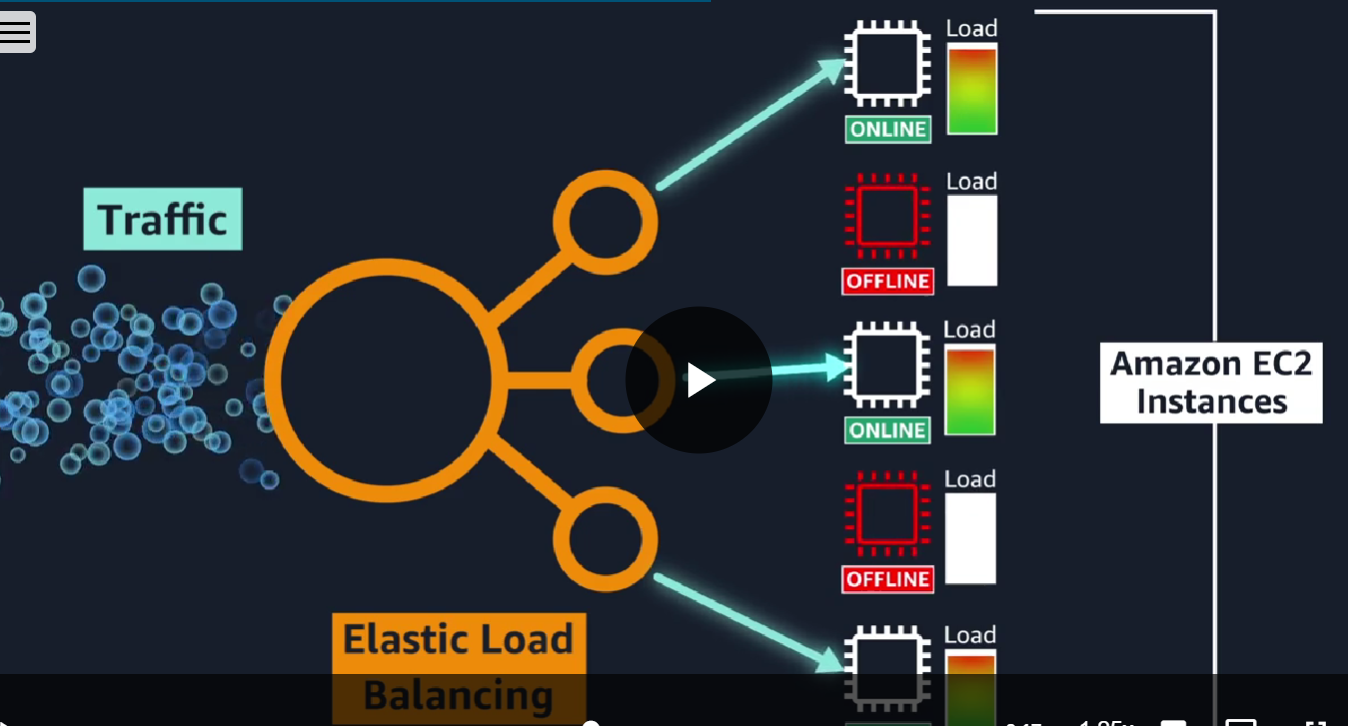
When configuring the size of your Auto Scaling group, you might set the minimum number of Amazon EC2 instances at one. This means that at all times, there must be at least one Amazon EC2 instance running.

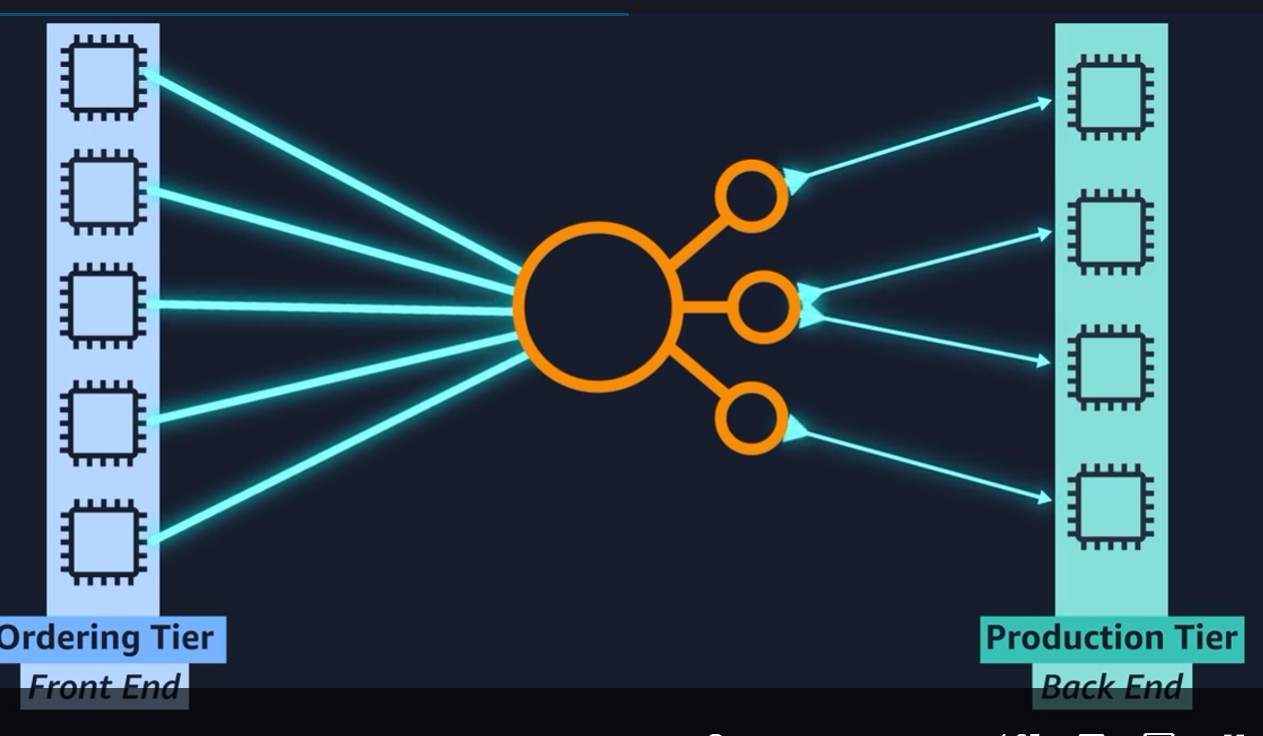
Next, you can set the **desired capacity** at two Amazon EC2 instances even though your application needs a minimum of a single Amazon EC2 instance to run.

The third configuration that you can set in an Auto Scaling group is the **maximum capacity**. For example, you might configure the Auto Scaling group to scale out in response to increased demand, but only to a maximum of four Amazon EC2 instances.

**Directing Traffic with Elastic Load Balancing**

**Elastic Load Balancing** is the AWS service that automatically distributes incoming application traffic across multiple resources, such as Amazon EC2 instances.  This **ELB** is automatically scalable.



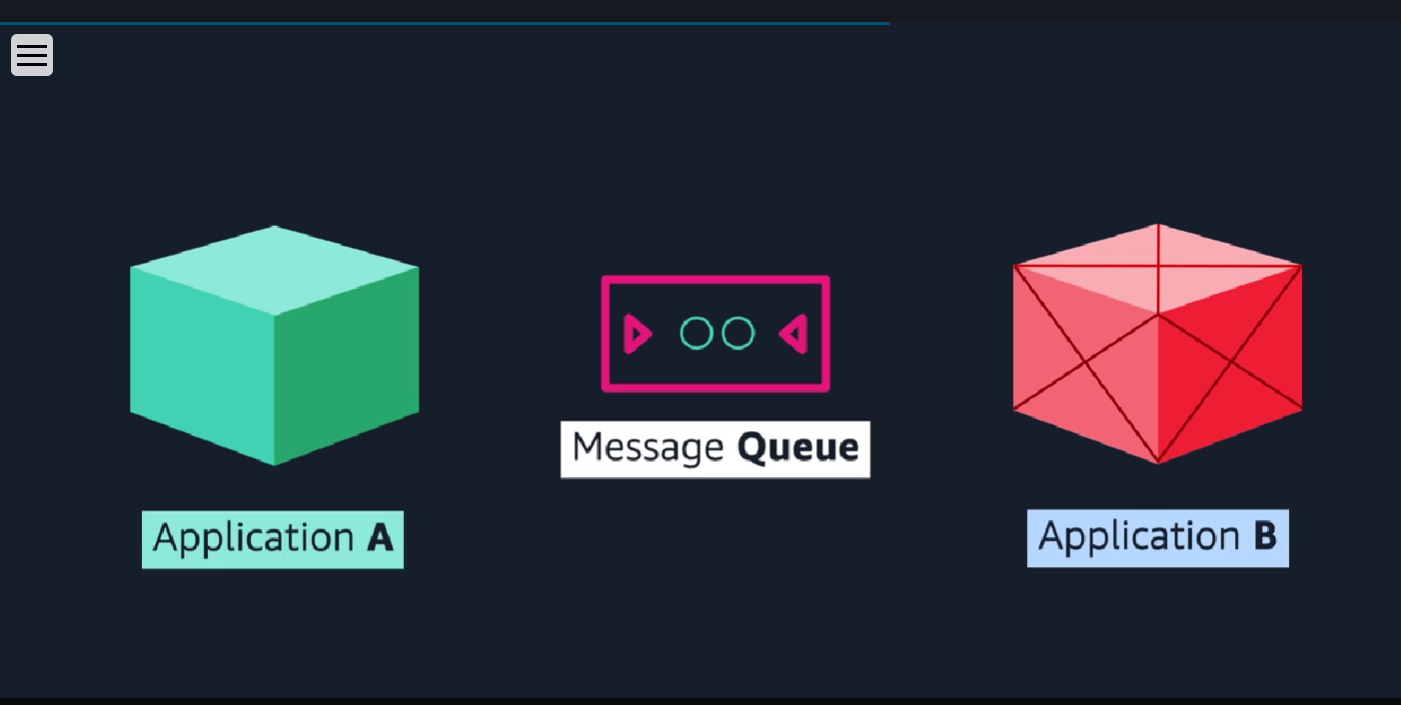


**Messaging and Queuing**

Instead of handing over the request directly to the backend, the frontend would post the order to some sort of buffer.  This idea of placing messages into a buffer is called **messaging and queuing.**

 If applications communicate directly like our cashier and barista previously, this is called being **tightly coupled**. if a single component fails or changes, it causes issues for other components or even the whole system.

A more reliable architecture is **loosely coupled**. This is an architecture where if one component fails, it is **isolated** and therefore won't cause cascading failures throughout the whole system. Messages are sent into the queue by Application A and they are processed by Application B. If Application B fails, Application A doesn't experience any disruption. Messages being sent can still be sent to the queue and will remain there until they are eventually processed.

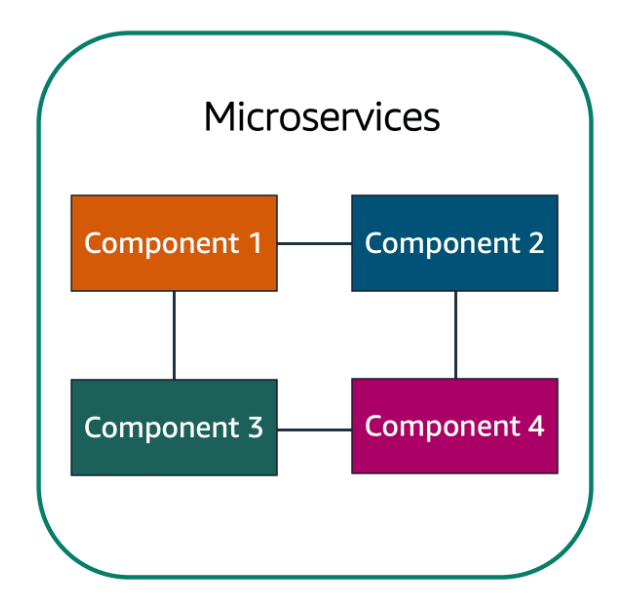


**2 common message/queue service by AWS: ---**

* **Amazon Simple Queue Service** or **SQS**
* **Amazon Simple Notification Service** or **SNS**.

To help maintain application availability when a single component fails, you can design your application through a **microservices** approach.

The loose coupling prevents the entire application from failing.



**Additional Compute Services**

**Serverless computing:**

To run in Amazon EC2, you must do the following:

* Provision instances (virtual servers).
* Upload your code.
* Continue to manage the instances while your application is running.

The term “**serverless**” means that your code runs on servers, but you do not need to provision or manage these servers.