**COMPUTE IN THE CLOUD**

***EC2***

EC2 runs on top of physical host machines managed by AWS using virtualization technology. When you spin up an EC2 instance, you aren't necessarily taking an entire host to yourself. Instead, you are sharing the host with multiple other instances, otherwise known as virtual machines.And a hypervisor running on the host machine is responsible for sharing the underlying physical resources between the virtual machines. This idea of sharing underlying hardware is called multitenancy. The hypervisor is responsible for coordinating this multitenancy and it is managed by AWS.

**Multitenancy**: Sharing underlying hardware between virtual machines

EC2 runs on top of physical host machines managed by AWS using virtualization technology.

EC2 instance, you can choose the operating system based on either Windows or Linux. You can provision thousands of EC2 instances on demand. With a blend of operating systems and configurations to power your business' different applications.

Beyond the OS, you also configure what software you want running on the instance. Whether it's your own internal business applications, simple web apps, or complex web apps, databases or third party software like enterprise software packages, you have complete control over what happens on that instance. EC2 instances are also resizable



Amazon Elastic Compute Cloud (Amazon EC2) provides secure, resizable compute capacity in the cloud as Amazon EC2 instances.

**Different types of EC2 instances**

* **General purpose**: General purpose instances provide a balance of compute, memory, and networking resources. You can use them for a variety of workloads, such as:application servers, gaming servers, backend servers for enterprise applications, small and medium databases

Suppose that you have an application in which the resource needs for compute, memory, and networking are roughly equivalent. You might consider running it on a general purpose instance because the application does not require optimization in any single resource area.

eg:**balances compute, memory, and networking resources**

* **Compute Optimised**: are ideal for compute-bound applications that benefit from high-performance processors. Like general purpose instances, you can use compute optimized instances for workloads such as web, application, and gaming servers.

However, the difference is compute optimized applications are ideal for high-performance web servers, compute-intensive applications servers, and dedicated gaming servers. You can also use compute optimized instances for **batch processing workloads** that require processing many transactions in a single group.

Eg: **high-performance processors**

* **Memory Optimised**: are designed to deliver fast performance for workloads that process large datasets in memory. In computing, memory is a temporary storage area. It holds all the data and instructions that a central processing unit (CPU) needs to be able to complete actions. Before a computer program or application is able to run, it is loaded from storage into memory. This preloading process gives the CPU direct access to the computer program.

Suppose that you have a workload that requires large amounts of data to be preloaded before running an application. This scenario might be a high-performance database or a workload that involves performing real-time processing of a large amount of unstructured data. In these types of use cases, consider using a memory optimized instance. Memory optimized instances enable you to run workloads with high memory needs and receive great performance.

Eg: **high-performance databases**

* **Accelerated Computing**: use hardware accelerators, or coprocessors, to perform some functions more efficiently than is possible in software running on CPUs. Examples of these functions include floating-point number calculations, graphics processing, and data pattern matching.

In computing, a hardware accelerator is a component that can expedite data processing. Accelerated computing instances are ideal for workloads such as graphics applications, game streaming, and application streaming.

* **Storage Optimised**:are designed for workloads that require high, sequential read and write access to large datasets on local storage. Examples of workloads suitable for storage optimized instances include distributed file systems, data warehousing applications, and high-frequency online transaction processing (OLTP) systems.

In computing, the term input/output operations per second (IOPS) is a metric that measures the performance of a storage device. It indicates how many different input or output operations a device can perform in one second. Storage optimized instances are designed to deliver tens of thousands of low-latency, random IOPS to applications.

Eg: **data warehousing applications**

**EC2 PRICING**

* **On Demand**:- On-Demand Instances are ideal for short-term, irregular workloads that cannot be interrupted. No upfront costs or minimum contracts apply. The instances run continuously until you stop them, and you pay for only the compute time you use.

Sample use cases for On-Demand Instances include developing and testing applications and running applications that have unpredictable usage patterns. On-Demand Instances are not recommended for workloads that last a year or longer because these workloads can experience greater cost savings using Reserved Instances.

* **Reserved instances**:These are suited for steady-state workloads or ones with predictable usage and offer you up to a 75% discount versus On-Demand pricing.Amazon EC2 pricing option provides a discount when you specify a number of EC2 instances to run a specific OS, instance family and size, and tenancy in one Region.
* **Savings Plan**:Savings Plan offers low prices on EC2 usage in exchange for a commitment to a consistent amount of usage measured in dollars per hour for a one or three-year term. This flexible pricing model can therefore provide savings of up to 72% on your AWS compute usage. This can lower prices on your EC2 usage, regardless of instance family, size, OS, tenancy, or AWS region. This also applies to AWS Fargate and AWS Lambda usage

*EC2 pricing option provides a discount when you make an hourly spend commitment to an instance family and Region for a 1-year or 3-year term*

* **Spot Instances:** Spot Instances are ideal for workloads with flexible start and end times, or that can withstand interruptions. Spot Instances use unused Amazon EC2 computing capacity and offer you cost savings at up to 90% off of On-Demand prices.eg: batch workloads.
* **Dedicated Hosts:** Dedicated Hosts are physical servers with Amazon EC2 instance capacity that is fully dedicated to your use. You can use your existing per-socket, per-core, or per-VM software licenses to help maintain license compliance. You can purchase On-Demand Dedicated Hosts and Dedicated Hosts Reservations. Of all the Amazon EC2 options that were covered, Dedicated Hosts are the most expensive.

**EC2 SCALING**

Amazon EC2 Auto Scaling enables you to automatically add or remove Amazon EC2 instances in response to changing application demand. By automatically scaling your instances in and out as needed, you can maintain a greater sense of application availability.

Within Amazon EC2 Auto Scaling, you can use two approaches: dynamic scaling and predictive scaling.

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

Suppose that you are preparing to launch an application on Amazon EC2 instances. 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.

When you create an Auto Scaling group, you can set the minimum number of Amazon EC2 instances. The **minimum capacity** is the number of Amazon EC2 instances that launch immediately after you have created the Auto Scaling group. In this example, the Auto Scaling group has a minimum capacity of one Amazon EC2 instance.

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**

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

A load balancer acts as a single point of contact for all incoming web traffic to your Auto Scaling group. This means that as you add or remove Amazon EC2 instances in response to the amount of incoming traffic, these requests route to the load balancer first. Then, the requests spread across multiple resources that will handle them. For example, if you have multiple Amazon EC2 instances, Elastic Load Balancing distributes the workload across the multiple instances so that no single instance has to carry the bulk of it.





**Messaging and Queuing**

Loosely coupled architecture, A single failure won't cause cascading failures. Message queue ie buffer between two applications.

**AMAZON SIMPLE QUEUE SERVICE(Amazon SQS) & AMAZON SIMPLE NOTIFICATION SERVICE(Amazon SNS)**

**Monolithic applications and microservices**

Suppose that you have an application with tightly coupled components. These components might include databases, servers, the user interface, business logic, and so on. This type of architecture can be considered a **monolithic application.**

In this approach to application architecture, if a single component fails, other components fail, and possibly the entire application fails.

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

In a **microservices approach**, application components are loosely coupled. In this case, if a single component fails, the other components continue to work because they are communicating with each other. The loose coupling prevents the entire application from failing.

In **Amazon SNS** is a publish/subscribe service. Using Amazon SNS topics, a publisher publishes messages to subscribers**.** subscribers can be web servers, email addresses, AWS Lambda functions, or several other options.

Using **Amazon SQS**, you can send, store, and receive messages between software components, without losing messages or requiring other services to be available.

**OTHER SERVICES:**

**Serverless**: you cannot see or access underlying infrastructure or instances that are hosting the application.Managing, scaling, provisioning, high availability and maintaining are done. 

***AWS Lambda***

Lambda's a service that allows you to upload your code into what's called a Lambda function. Configure a trigger and from there, the service waits for the trigger. When the trigger is detected, the code is automatically run in a managed environment, an environment you do not need to worry too much about because it is automatically scalable, highly available and all of the maintenance in the environment itself is done by AWS



**\*Amazon elastic container service(amazon ECS)**

Amazon Elastic Container Service (Amazon ECS) is a highly scalable, high-performance container management system that enables you to run and scale containerized applications on AWS.

Amazon ECS supports Docker containers. Docker is a software platform that enables you to build, test, and deploy applications quickly. AWS supports the use of open-source Docker Community Edition and subscription-based Docker Enterprise Edition. With Amazon ECS, you can use API calls to launch and stop Docker-enabled applications.

**\*Amazon elastic kubernetes service (amazon EKS)** these are container orchestration tools, here container is docker container. Container is a package of code.

Amazon Elastic Kubernetes Service (Amazon EKS) is a fully managed service that you can use to run Kubernetes on AWS.

Kubernetes is open-source software that enables you to deploy and manage containerized applications at scale. A large community of volunteers maintains Kubernetes, and AWS actively works together with the Kubernetes community. As new features and functionalities release for Kubernetes applications, you can easily apply these updates to your applications managed by Amazon EKS.

**\*AWS Fargate**

AWS Fargate is a serverless compute engine for containers. It works with both Amazon ECS and Amazon EKS.

When using AWS Fargate, you do not need to provision or manage servers. AWS Fargate manages your server infrastructure for you. You can focus more on innovating and developing your applications, and you pay only for the resources that are required to run your containers.

if you don't want to even think about using EC2s to host your containers because you either don't need access to the underlying OS or you don't want to manage those EC2 instances, you can use a compute platform called **AWS Fargate**

**If you are trying to host traditional applications and want full access to the underlying operating system like Linux or Windows, you are going to want to use EC2. If you are looking to host short running functions, service-oriented or event driven applications and you don't want to manage the underlying environment at all, look into the serverless AWS Lambda. If you are looking to run Docker container-based workloads on AWS, you first need to choose your orchestration tool. Do you want to use Amazon ECS or Amazon EKS? After you choose your tool, you then need to choose your platform. Do you want to run your containers on EC2 instances that you manage or in a serverless environment like AWS Fargate that is managed for you**