



Cloud Computing

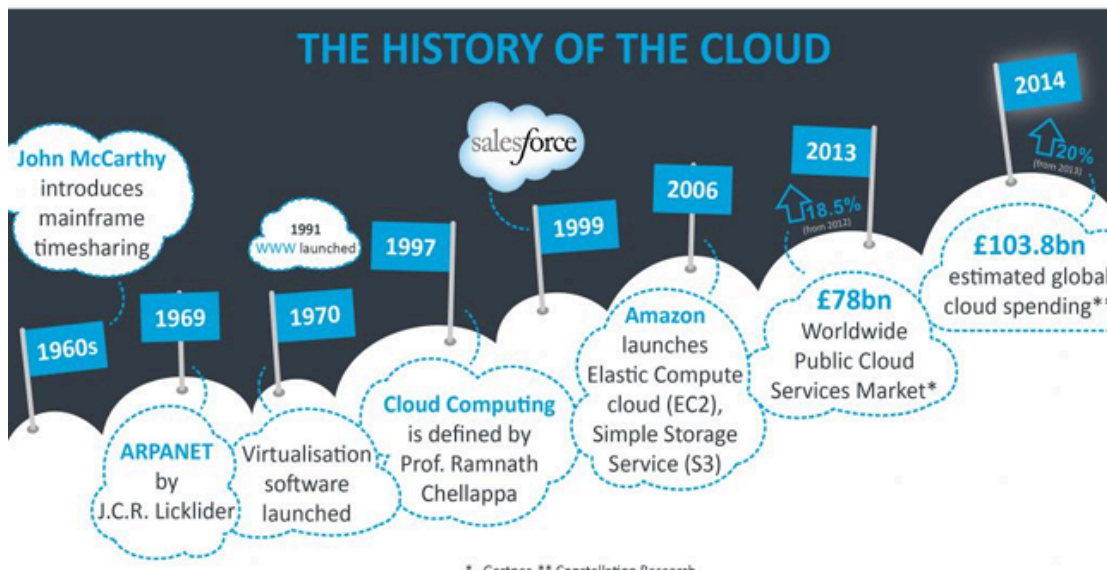
Cloud Computing

Cloud computing is like renting a computer and storage space on the internet instead of buying and maintaining your own.

In cloud computing, companies like **Google, Amazon, and Microsoft** own massive computers (servers) and you can use them over the internet to store files, run applications, or even process data without having to set up your own hardware.

Characteristics Of Cloud Computing

- Scalability (It Grows or Shrinks as Needed) (aka Rapid Elasticity)
- Resource pooling (Sharing Like a Library with location independence)
- Pay-per-use pricing (Only Pay for What You Use)
- Resilience & Availability (Always Online) (Cloud services are **available 24/7**, and even if one server crashes, another one takes over.)
- Measured Service (Tracking Usage Automatically)
- Broad Network Access (Accessible from Anywhere and from any device)
- Security (If you lose your phone, your Google Drive files are still safe because they are stored in the cloud, not on your device.)
- Automation (Think of Google Photos automatically backing up your pictures. Cloud services can automate tasks like updates, security checks, and scaling resources without human intervention.)
- Cost Saving (No need of hardware)
- Easy Maintenance



Cloud Infrastructure

- The first thing needed for cloud computing is **infrastructure**—big, powerful computers (servers) that store data and run programs.
- These servers can be **virtualized (shared)** or **bare metal (dedicated to one user)**.
- **Example:**
 - Think of a **hotel**. It has many rooms (resources like storage and computing power).
 - Some rooms are **shared (virtualized)** like a dormitory, while others are **private (bare metal servers)** like luxury suites.

Cloud Services (What You Can Get in the Cloud)

Cloud computing offers three main types of services:

1. Infrastructure as a Service (IaaS) – Renting Raw Materials

- You get **basic computing resources** like storage, virtual machines, and networks.

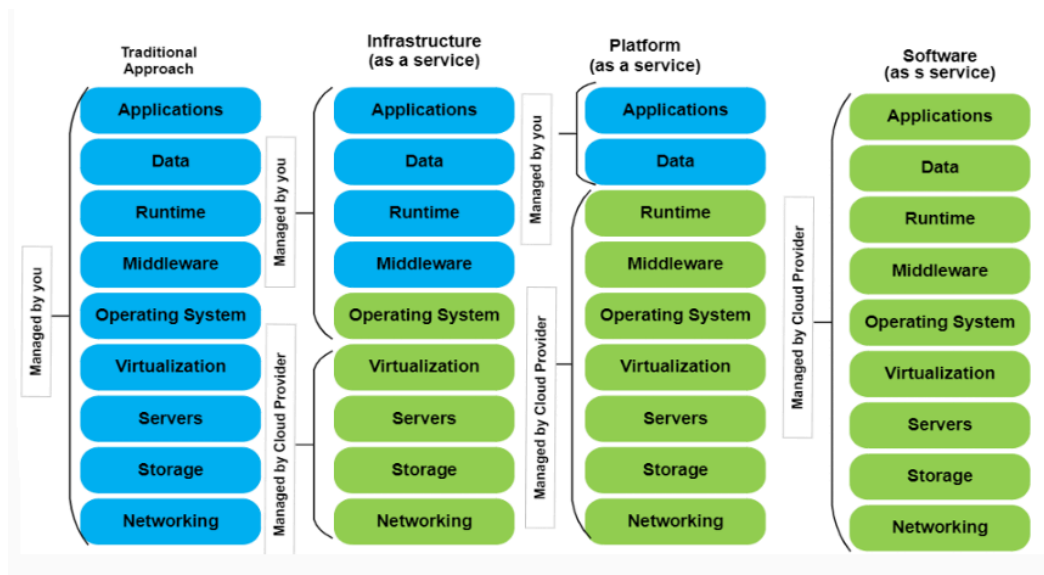
- Example: **AWS EC2**, where you rent virtual machines.
- Used for migrating workloads (existing applications) to the cloud
- Used for hosting your website
- Also used for storage of data , backup and recovery.
- **Analogy:** Renting a piece of land and building your own house from scratch.

2. Platform as a Service (PaaS) – Ready-Made Development Tools

- You get a **development platform** with tools and a managed environment.
- Example: **Google App Engine**, Databases where you can build apps without worrying about infrastructure.
- **Analogy:** Getting a **fully equipped kitchen** where you just cook your meal instead of buying all the tools separately.
- Automatic backup of data

3. Software as a Service (SaaS) – Ready-to-Use Software

- You use **fully developed applications** hosted in the cloud.
- Example: **Gmail, Dropbox, Microsoft 365.**
- **Analogy:** Instead of cooking, you order food from a restaurant (**ready-made service**)



Cloud Service	Definition	Analogy (Real-Life Example)	Key Difference
Network as a Service (NaaS)	Provides virtual network services like VPNs, firewalls, and bandwidth on demand without needing physical network hardware.	Renting internet access instead of setting up your own network infrastructure. Example: Using a VPN service instead of setting up your own secure network.	Removes the need for companies to buy expensive networking equipment.
Desktop as a Service (DaaS)	Provides a virtual desktop environment where users can access their desktop from anywhere.	Using a powerful PC remotely from a weak laptop or tablet. Example: Accessing a Windows desktop from an iPad.	No need to buy expensive computers; the cloud provider manages everything.
Storage as a Service (STaaS)	Offers cloud-based storage where users can store and access files online.	Using a cloud drive instead of buying a physical hard disk. Example: Google Drive or Dropbox.	Users pay for storage space instead of maintaining their own servers.
Database as a Service (DBaaS)	Provides cloud-based databases where	Renting a database instead of installing one	Database maintenance,

	users can store, manage, and access data without handling setup or maintenance.	on your own server. Example: MongoDB Atlas or Amazon RDS.	security, and scaling are handled by the cloud provider.
Data as a Service (DaaS)	Provides access to data (like weather reports, maps, or stock prices) through cloud APIs.	Renting data instead of collecting and storing it yourself. Example: Google Maps API for location services.	Users can access real-time, high-quality data without managing it themselves.
Security as a Service (SECaaS)	Provides security services like antivirus, firewall, and intrusion detection through the cloud.	Hiring security guards instead of securing your home yourself. Example: McAfee Cloud Security or Cisco Security.	Protects businesses from cyber threats without requiring in-house security experts.
Identity as a Service (IDaaS)	Manages authentication and user identity, enabling single sign-on (SSO) and access control.	Using Google or Facebook to log in to multiple websites instead of creating separate accounts.	Helps organizations manage logins securely without storing passwords themselves.

Operating System Service (The Manager of the Cloud)

- This is the **software that runs the cloud services**.
- It handles the **servers, networks, and user requests**.
- **Example:**
 - Just like a hotel **staff manages rooms**, the Operating System Service **manages cloud resources** for users.

Business System Service (BSS) – The Billing & Validation System

- This service checks **if a user is eligible** to access cloud resources and generates invoices.
- It considers factors like:

- Number of users
- CPU power
- Memory
- Storage
- Hours used per month
- **Example:**
 - Just like **Netflix bills you based on your subscription plan**, the cloud provider **bills based on your usage**.

Final Analogy

- **Infrastructure = The hotel itself** (Rooms available for rent).
- **Cloud Services = Different types of rooms** (Standard room = IaaS, Suite with facilities = PaaS, All-inclusive stay = SaaS).
- **Operating System Service = The hotel staff** (Manages everything).
- **Business System Service = The hotel reception** (Checks bookings, validates payment, and issues invoices).
- **Monthly Invoice = Your hotel bill** (Charges you based on the services you used).

Deployment Models

Cloud Type	Who Uses It?	Example	Analogy
Public Cloud	Open for anyone	Gmail, Google Drive, AWS, Netflix	A public gym where anyone can join
Private Cloud	Used by one company	Bank's private data center	A home gym for a single family
Hybrid Cloud	Mix of private + public	E-commerce sites storing private data	Home gym + Public gym combo

		but using public servers for traffic	
Community Cloud	Shared by multiple organizations with similar needs	Government agencies or hospitals sharing cloud storage	A private gym shared by hospitals or universities
Multi-Cloud (Using Many Clouds Together)	Using multiple cloud providers (AWS, Google Cloud, Azure) for different services.	A company uses AWS for storage, Google Cloud for AI, and Azure for security.	Eating at different restaurants: Choosing McDonald's for burgers, Starbucks for coffee, and Domino's for pizza—using different services for different needs.

P.S: Community Cloud is more secure than Public Cloud and more cost effective than private cloud.

Characteristics	Public Cloud	Private Cloud	Community Cloud	Hybrid Cloud
Demand for in-house infrastructure	Not required	Required for private cloud	Shared among organizations	Mandatory
Ease of use	Very easy to use	Requires an operational IT staff	Requires an operational IT staff from multiple organizations	Complex because it involves more than one deployment model
Cost	Affordable and lower compared to other models	High compared to public cloud	Cost is distributed among organizations	Cheaper than private cloud and costlier than public cloud

Security	Less secure than other models	Provides more security than public cloud	Higher than public cloud and lower than private cloud	Higher than public cloud and lower than private and community cloud
Ownership	Cloud service provider	Single organization	Multiple organizations with similar concerns	Cloud service provider for public cloud and organization for private cloud
Managed by	Cloud service provider	Organization operational staff	Operational staff among multiple organizations	Cloud service provider for public cloud and operational staff for private cloud
Scalability	Very High	High	Fixed	High
Reliability	Low	Low	High	High
Data Privacy	Low	High	High	High

Disadvantages of Cloud

- ✗ **Internet Dependency** – You need an internet connection.
- ✗ **Vendor Lock-In** – Hard to switch between different cloud providers.
- ✗ **Limited Control** – The cloud provider manages everything, so you don't have full control.
- ✗ **Security Concerns** – Risk of data breaches and unauthorized access
- ✗ **Compliance Issues** – Challenges in meeting industry-specific regulations

Virtualization

Virtualization is **the technology that makes cloud computing possible**. It allows one physical computer to run multiple virtual machines.

Virtualization is basically Creating multiple virtual instances on a single physical machine.

◆ Hypervisor (The Manager of Virtual Machines)

- The software that creates and manages virtual machines is called a **Hypervisor**.

Cloud Service Providers

- **Amazon Web Services (AWS)** – The largest cloud provider.
- **Microsoft Azure** – Popular for enterprise and business solutions.
- **Google Cloud Platform (GCP)** – Best for AI and machine learning.
- **IBM Cloud** – Focuses on AI and enterprise solutions.
- **Oracle Cloud** – Best for databases and enterprise software.

Virtualization Concepts

Term	Definition	Example	Analogy
Virtual Machine (VM)	A software-based computer that acts like a real one, running its own OS and applications.	Running Windows inside a Mac using VMware or VirtualBox.	A rental apartment inside a big house: Each tenant (VM) has their own setup but shares the same building (physical computer).
Hypervisor	A software layer that creates and manages multiple VMs on a single physical machine.	VMware ESXi, Microsoft Hyper-V, VirtualBox.	A landlord managing multiple apartments: Ensures that each tenant (VM) gets their fair share of electricity, water, and space (RAM, CPU, storage).
Virtual Machine Monitor (VMM)	A software component that manages interactions	KVM (Kernel-based Virtual Machine) in Linux.	A security guard in an apartment complex: Ensures tenants (VMs)

	between the VMs and the host machine.		follow rules and don't interfere with each other.
--	---------------------------------------	--	---

AWS Services Cheat Sheet

Category	Service	Purpose	Example
Compute	EC2	Virtual machines in the cloud	Hosting a website
	Lambda	Serverless computing	Running code without managing servers
Storage	S3	Object storage for files, images	Storing backups, website assets
	EBS	Block storage for EC2 instances	Virtual hard disk for EC2
Database	RDS	Managed relational database	MySQL, PostgreSQL, Oracle
	DynamoDB	NoSQL database	Scalable key-value store
Networking	VPC	Private cloud network	Securely connect cloud resources
	Route 53	DNS service	Managing website domain names
Security	IAM	Identity management	User authentication & permissions
	WAF	Web Application Firewall	Protects against cyber attacks
Monitoring	CloudWatch	Monitor AWS resources	Performance tracking
	CloudTrail	Log AWS API activity	Security auditing

Amazon Web Services (AWS)

AWS is like a **supermarket for computing** – you can pick and choose the services you need.

1. Compute (Processing Power) – EC2 (Elastic Compute Cloud)

- **What is EC2?** A **virtual computer** (VM) running in the cloud.
 - **Example:** Instead of buying a laptop, you rent a high-performance computer online.
 - **Types of EC2 Instances:**
 - **On-Demand** – Pay as you go. **(Like booking a hotel room for a night)**
 - **Reserved Instances** – Pre-pay for 1 or 3 years. **(Like renting an apartment for a long term)**
 - **Spot Instances** – Use unused cloud capacity at a lower price. **(Like bidding for last-minute flight tickets)**
-

2. Storage – Amazon S3 (Simple Storage Service)

- **What is S3?** A place to store files, photos, and videos online.
 - **Example:** Like **Google Drive or Dropbox**, but more powerful and scalable.
-

3. Databases – Amazon RDS (Relational Database Service)

- **What is RDS?** A **managed** database service where AWS handles backups and maintenance.
 - **Example:** Instead of setting up a database yourself, AWS does it for you.
 - **Analogy:** Like using **Google Docs** instead of installing Microsoft Word.
-

4. Networking – Virtual Private Cloud (VPC)

- **What is VPC?** A **private network inside AWS** where your resources (servers, storage) live.
 - **Analogy:** Like having a **private WiFi network** at home instead of using public WiFi.
-

5. Identity & Security – IAM (Identity and Access Management)

- **What is IAM?** A tool to **control who can access what** in AWS.
 - **Example:** Like giving different keys to employees in an office – some can enter all rooms, some only the main entrance.
-

6. Load Balancing & Auto Scaling (Handling Website Traffic)

- **Load Balancer:** Spreads traffic across multiple servers to prevent overload.
 - **Auto Scaling:** Automatically adds or removes servers based on traffic.
 - **Example:**
 - **Without Auto Scaling:** A pizza shop hires 2 chefs permanently, but struggles when 50 orders come in.
 - **With Auto Scaling:** More chefs (servers) are hired when orders increase and sent home when orders decrease.
-

How to Use AWS for Free?

- AWS **Free Tier** allows **750 hours of free usage** per month.
 - You can create a **virtual machine (EC2 instance)**, store files (S3), and use databases for free.
-

Cloud Computing Architecture

1 Event-Driven Architecture (EDA) → Instant Reaction System

- Works like **WhatsApp notifications** – reacts instantly when something happens.
- Example: **Food delivery apps** send real-time updates when your order is placed, prepared, or delivered.
- **Why?** Improves speed, scalability, and efficiency.

2 Service-Oriented Architecture (SOA) → Reusable Service System

- Works like a **Lego set** – uses **pre-built services** to build applications faster.
- Example: **Amazon** reuses payment services (PayPal), shipping services (DHL), and customer support chatbots.
- **Why?** Saves time, money, and makes apps easier to maintain.

3 Service Level Agreement (SLA) → Cloud Contract

- A contract between the **cloud provider and customer** that defines rules.
- Example: **Netflix promises 99.9% uptime** – if the service fails, they may offer a refund.
- **Why?** Ensures quality, security, and reliability.

◆ **Final Analogy → Cloud = A Food Delivery App**

- ✓ **EDA** = Real-time tracking notifications.
- ✓ **SOA** = Different services working together (payment, delivery, customer support).
- ✓ **SLA** = Guarantee of quality service (fast delivery, fresh food).

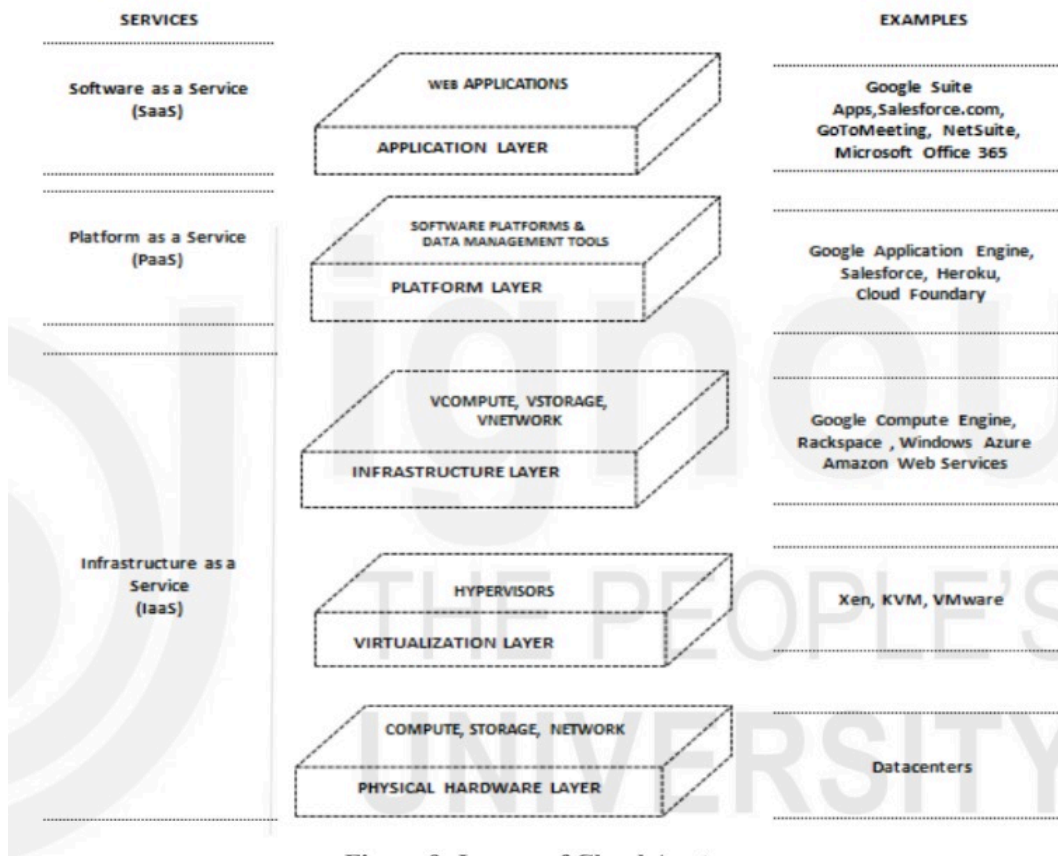


Figure 9: Layers of Cloud Anatomy

Cloud Layer	Definition	Real-Life Analogy	Purpose
-------------	------------	-------------------	---------

Client Access Layer	The topmost layer where users interact with the cloud using devices like smartphones, laptops, and tablets.	Front Door of an Apartment – This is where residents (users) enter the building (cloud).	Allows users to connect to the cloud using different devices.
Internet Connectivity Layer	Connects users to the cloud using the internet. Public clouds use the internet, while private clouds may use local networks.	Elevator or Staircase – You need an elevator to reach your apartment, just like you need the internet to access the cloud.	Ensures users can reach cloud services via the internet.
Cloud Service Management Layer	Manages the cloud, including resources, scheduling, and security. It follows rules set in Service Level Agreements (SLAs) between the provider and users.	Building Management Office – Ensures the apartment is well-maintained and services like security, repairs, and utilities work properly.	Manages cloud resources, ensures smooth operations, and enforces SLAs.
Physical Resources Layer	The foundation of the cloud, consisting of real servers, storage devices, and networking hardware located in data centers.	Building Foundation (Concrete, Pipes, Electricity) – Without a strong foundation, the building wouldn't exist.	Provides the necessary physical hardware to run cloud services.

Cloud Anatomy vs. Cloud Architecture

Now, let's differentiate **Cloud Anatomy** from **Cloud Architecture** using another analogy:

- **Cloud Architecture** is **how the apartment is designed**, including how rooms are arranged, water pipes are placed, and electrical wiring is installed.
- **Cloud Anatomy** is **what's inside the apartment**, like furniture, appliances, and decorations.

Cloud Anatomy Layer	Definition	Real-Life Analogy
Application Layer	The topmost layer where software applications run.	Furniture & Decor – What people see and use daily (e.g., chairs, tables, decorations).
Platform Layer	Provides tools and environments for developers to build and run applications.	Kitchen Setup – Provides appliances and tools to cook meals (build apps).
Infrastructure Layer	Provides computational resources like virtual machines and networking.	Electricity & Water Supply – Essential resources that keep the house running.
Virtualization Layer	Creates multiple virtual machines using a single physical server.	Apartment Partitioning – A single apartment is divided into multiple smaller rooms for different people.
Physical Hardware Layer	The actual hardware (servers, databases, and networking devices) that powers the cloud.	Building Materials (Bricks, Steel, Cement) – The physical foundation of the apartment.

TTM

Time to Market (TTM) is the total time it takes from the moment you think of the cake idea to the day you actually start selling it in a bakery.

Cloud Scalability

Scalability Type	Definition	How It Works	When to Use?	Example in Cloud
Vertical Scaling (Scale-Up/Down)	Increasing or decreasing the power of an existing machine (CPU, RAM, Storage).	Upgrade to a more powerful server or downgrade when not needed.	When your application needs more power but doesn't require multiple servers.	Upgrading an AWS t2.micro instance to t2.large for better performance.

Horizontal Scaling (Scale-Out/In)	Adding or removing multiple machines to distribute the load.	New instances (servers) are added or removed dynamically based on demand.	When traffic fluctuates and a single machine can't handle it.	AWS Auto Scaling Groups adding EC2 instances when website traffic spikes.
Diagonal Scaling (Hybrid Scaling)	A combination of vertical and horizontal scaling for maximum efficiency.	First, increase the power of an existing machine, then add more servers if needed.	When both performance boost and distributed load handling are required.	Start with a t2.large EC2 instance, then add more instances via load balancing .
Auto-Scaling	Automatically adjusts resources based on demand.	Cloud provider scales up or down without manual intervention .	When traffic is unpredictable , such as seasonal spikes.	AWS Auto Scaling + Load Balancer adjusting instances based on website traffic.

P.S: **Diagonal Scaling (Hybrid of Vertical & Horizontal)**

◆ **What it means?** First, scale **vertically** (upgrade the machine), and when that maxes out, **scale horizontally** (add more machines).

EC2 Instances

EC2 is like renting a virtual computer in the cloud. Instead of buying and maintaining physical servers, you can **launch, use, and stop virtual machines** (called **instances**) anytime you want.

Types of EC2 Instances

- 1 **General Purpose** – Balanced for websites, small databases.
- 2 **Compute-Optimized** – Best for high-performance computing & AI.
- 3 **Memory-Optimized** – For large databases and real-time analytics.

4 **Storage-Optimized** – For apps needing high-speed data access.

5 **Accelerated Computing** – Uses GPUs for heavy graphics & AI tasks.

Amazon Machine Image (AMI)

Feature	Explanation
What is AMI?	A pre-configured template (blueprint) to create EC2 instances.
Why Use AMI?	Saves time by allowing you to launch multiple identical servers quickly.
Key Components	- OS & Software Configuration - Launch Permissions - Storage Setup
How to Use?	- Create an AMI from an existing EC2 instance. - Use the AMI to launch multiple instances. - Share, sell, or keep it private.
Types of AMI	- Amazon AMIs (Default OS images) - Marketplace AMIs (Paid, third-party) - Community AMIs (Free, shared) - Custom AMIs (User-created)
Pricing	- Public AMI = Free - Private AMI = Paid (depends on seller) - Costs vary by region & data transfer
Example Use Case	Need 10 servers with the same setup? Create one EC2 instance → Save as AMI → Launch 10 copies instantly!

Key Components of AMI

1. Root Volume (Storage for the OS)

- Think of this as the **hard drive** of your virtual machine.
- It contains the **operating system** and important files needed to boot the server.
- **Two types of root volumes:**
 - **Instance Store:** Temporary storage (data is lost if the instance stops).
 - **Amazon EBS (Elastic Block Store):** Persistent storage (data is saved even if the instance stops).

2. Launch Permissions

- Controls **who** can use the AMI to create instances.

- **Three types:**

- **Public:** Anyone can use it.
- **Explicit:** Only specific AWS accounts can use it.
- **Implicit:** Only the owner (you) can use it by default.


3. Block Device Mapping

- Determines **which storage volumes (hard drives)** are attached to the instance when it starts.


Types of AMIs (Different AMI Options in AWS)

There are **4 types** of AMIs in AWS:


1 Amazon-provided AMIs (AWS Default)

- These are **official AMIs from AWS** with basic operating systems (Amazon Linux, Windows, etc.).
-  **Best for:** Quick setups with standard OS.

2 Marketplace AMIs (Paid AMIs)


- These are **AMIs from third-party vendors** in AWS Marketplace.
- They come with **pre-installed software** but may require a license fee.
-  **Best for:** Deploying enterprise software (e.g., SAP, Windows Server with SQL).

3 Community AMIs (Free Public AMIs)

- Created by other AWS users and **shared with the public**.
- They may not always be secure or up-to-date.
-  **Best for:** Testing purposes (but be careful when using them).

4 Custom AMIs (Your Own AMI)

- You can **create your own AMI** after setting up an EC2 instance exactly how you want it.

- This is great when you need to **replicate custom configurations across multiple instances**.
 -  **Best for:** Scaling your business with **pre-configured environments**.
-

Amazon EC2

EC2 stands for Elastic Compute Cloud

Amazon Elastic Compute Cloud (EC2) is like renting a computer in the cloud instead of buying one. It lets you increase or decrease computing power (CPU, RAM, storage, bandwidth, etc.) whenever you need it. You pay only for what you use—just like paying for electricity instead of owning a power plant.

Key Features of EC2 :

- **Scalability** – You can increase or decrease the computer power anytime.
- **Pay-as-You-Go** – You only pay for what you use, just like paying an electricity bill.
- **Multiple Regions** – Amazon EC2 is available in 22 global locations, so you can choose the nearest one for faster performance.
- **Security** – It provides encryption and firewalls to keep your data safe.

Instance: A virtual machine (VM) running on the cloud, like a mini computer.

AMI (Amazon Machine Image): A template for launching a new instance (like installing an OS on your computer).

EBS (Elastic Block Storage): Cloud-based storage for EC2 instances.

Types of EC2 Pricing:

1. On-Demand Instances

- Pay per hour. No long-term commitment.
- Example: If you need a server for one-time use, you can rent it for a few hours.

2. Reserved Instances

- Pay upfront for long-term use, cheaper than On-Demand.
- Example: If a company needs EC2 for a whole year, it can buy a Reserved Instance to save money.

3. Spot Instances

- Buy unused EC2 capacity at a discounted price (up to 90% off).
- Example: If Amazon has extra servers not in use, you can rent them at a lower price.

4. Savings Plans

- Commit to using EC2 for 1 or 3 years at a discount.
- Example: If you know you'll need EC2 for a long time, this saves money.

AWS Regions, Availability Zones, and Points of Presence

Term	Definition	Example	Analogy
Region	A geographic location with multiple AWS data centers.	"US-East-1" in Virginia.	Different bank branches in different cities.
Availability Zone (AZ)	A data center inside a region.	Virginia has 3 AZs.	Each bank branch has multiple cash counters.
Point of Presence (PoP)	Edge locations to speed up content delivery.	AWS CloudFront (CDN).	Amazon warehouses in different cities to speed up delivery.

EC2 Linux Instance Using PuTTY?

Imagine you have a computer (server) running Linux on AWS (Amazon Web Services), and you want to control it from your Windows PC. You need a way to connect to this remote Linux machine securely. That's where **PuTTY** comes in—it's a tool that lets you access your AWS Linux machine from your Windows PC.

Understanding the Key Files

1. .pem (Privacy Enhanced Mail) → Think of it like a special **key** given by AWS to unlock your Linux machine. It's a security file that AWS provides when you create an EC2 instance.
2. .ppk (PuTTY Private Key) → PuTTY, the tool we use on Windows, **doesn't understand** .pem files directly. So, we need to convert the .pem key into a .ppk key, which PuTTY can use.

Default Port for SSH : 22

whoami → shows the current user (ec2-user is the default user for AWS Linux)

What is Amazon Glacier?

Amazon Glacier is a cloud storage service provided by AWS (Amazon Web Services) that is designed for storing data that you don't need to access frequently. It's mainly used for long-term storage, backups, and archives. Think of it like a super cheap digital attic where you can store old files safely but retrieving them takes some time.

◆ **Example:** Imagine you have a company that collects customer data, logs, or old videos that you don't need daily but must keep for years (e.g., legal documents, medical records, or security footage). Instead of storing this data on an expensive, fast-access cloud service, you can use Amazon Glacier, which is much cheaper but takes longer to retrieve files.

Key Differences Between Amazon S3 and Amazon Glacier

AWS has many storage options. The two most common ones are:

1. **Amazon S3** – Fast and good for frequently accessed data.
2. **Amazon Glacier** – Slow retrieval, but much cheaper, ideal for long-term storage.

📌 **Example:**

- If you run a website with product images that users access daily, use **Amazon S3**.

- If you have old company reports that you might need once a year, store them in **Amazon Glacier**.

Feature	Amazon S3 (Frequent Storage)	Amazon Glacier (Long-term)
Use Case	Daily access to files	Rarely accessed data
Storage Cost	Higher	Lower
Retrieval Speed	Instant	Takes minutes to hours
Best For	Websites, active databases	Backups, compliance storage

How Does Amazon Glacier Store Data?

Amazon Glacier organizes data into **Archives** and **Vaults**:

1. **Archives** – Individual files stored in Glacier. Can be documents, videos, or backups.
2. **Vaults** – Storage containers for organizing multiple archives.

 **Example:** Think of a vault as a **cabinet**, and each archive inside it is a **file folder**.

Because Glacier is for rarely accessed data, getting your files back isn't instant. There are three options:

1. **Expedited Retrieval (1-5 minutes)** – If you need data urgently, you pay extra for faster access.
2. **Standard Retrieval (3-5 hours)** – Cheaper but takes a few hours.
3. **Bulk Retrieval (5-12 hours)** – Cheapest but takes the longest.

What is Amazon S3?

AWS S3 is a cloud storage service provided by Amazon Web Services (AWS). It is used for:

- ✓ Storing files securely
- ✓ Retrieving files whenever needed

✓ Keeping data safe and backed up

Think of it like **Google Drive or Dropbox** but designed for businesses, websites, and large applications.

An **S3 Bucket** is like a **folder** where you store files in AWS S3.

- Every file in S3 is called an **Object** (like a document or image).
- Each object has a **key** (unique name).
- You can set permissions on who can access the bucket.

Key Features of Amazon S3

- 1 **Versioning** – Keeps multiple versions of a file, preventing accidental deletion.
- 2 **Access Control Lists (ACLs)** – Manage who can view or edit files.
- 3 **Bucket Policies** – Set rules for user access.
- 4 **Lifecycle Rules** – Move old files to cheaper storage (Glacier) or delete them after a time.


📌 **Example:** If you mistakenly delete an important file, versioning allows you to recover it.

Types of S3 Storage Classes

AWS S3 offers different storage types based on cost and access needs:


Storage Class	Best For	Cost	Example
Standard	Frequently accessed files	High	Daily-use images, videos
Standard IA	Infrequent access	Lower	Backup logs, archives
Intelligent Tiering	Auto-moves files between Standard & IA	Moderate	Unpredictable data usage
One Zone IA	Single region storage	Lower	Temporary backup files

Glacier	Long-term storage	Very low	Old data, compliance records
----------------	-------------------	----------	------------------------------

 **Example:** If you store old tax documents that you rarely need, you can use **S3 Glacier** to save costs

AWS S3 vs. Google Drive vs. Dropbox

Feature	AWS S3	Google Drive	Dropbox
Usage	Large-scale cloud storage	Personal storage	Personal/business storage
Scalability	Extremely high	Medium	Medium
Pricing	Pay-as-you-go	Free up to 15GB	Free up to 2GB
Best For	Developers, Businesses	Individuals, Students	Small Teams

 **Example:** A startup can use AWS S3 to store millions of customer records, while an individual might use Google Drive for personal documents.

AWS Auto Scaling

AWS **Auto Scaling** works just like this! It **automatically adds more computing resources (like servers) when traffic is high and removes them when traffic is low** to save money while keeping performance stable.

How AWS Auto Scaling Works

Let's break it down into simple steps:

1. **Monitoring Demand** – AWS keeps an eye on your application's traffic and resource usage (CPU, memory, etc.).
2. **Scaling Up (Adding More Servers)** – If traffic increases, AWS automatically adds more servers to handle the load.
3. **Scaling Down (Removing Extra Servers)** – When traffic decreases, AWS removes the extra servers to save costs.

4. **Load Balancing** – Think of a restaurant where customers are directed to the least busy cashier. Similarly, AWS ensures that users are directed to available servers, improving performance.
5. **Cost Optimization** – Since AWS only adds resources when needed, you only pay for what you use, reducing costs.

Benefits of Auto Scaling

1. **No Manual Work** – You don't have to manually add or remove servers; AWS does it for you.
2. **Saves Money** – You only pay for the servers you actually use.
3. **Better Performance** – Your application remains fast even during high traffic.
4. **Prevents Downtime** – Your app doesn't crash because of sudden traffic spikes.

Types of Auto Scaling

AWS provides different ways to scale your application:

1. **Dynamic Scaling (Real-time Adjustments)**
 - Automatically increases or decreases servers **based on live traffic**.
 - Example: A news website gets high traffic when breaking news is published. Auto Scaling adds servers instantly.
2. **Predictive Scaling (Future Planning)**
 - Uses **historical data** to predict future traffic and prepares resources in advance.
 - Example: An e-commerce website gets high traffic every **Friday night**. AWS scales up in advance.
3. **Scheduled Scaling (Fixed Schedule)**
 - You define a specific time to scale up or down.
 - Example: A school's online learning platform scales up **every morning** when students log in and scales down in the evening.
4. **Manual Scaling (Human Control)**

- You manually decide when to increase or decrease servers.
- Example: A company manually adds servers before launching a marketing campaign.

Amazon EC2 Auto Scaling

Amazon EC2 (Elastic Compute Cloud) provides virtual servers to run your applications. **EC2 Auto Scaling** manages these servers by:

- Adding new EC2 instances when traffic increases.
- Removing extra EC2 instances when traffic decreases.
- Ensuring your app always runs smoothly without wasting money.

Key Components of Auto Scaling

1. **Auto Scaling Groups** – A collection of EC2 instances managed together.
2. **Launch Templates** – A blueprint defining how to create new EC2 instances.
3. **Scaling Policies** – Rules that determine when AWS should scale up or down.

Scaling Strategies

AWS offers different ways to manage Auto Scaling:

1. Horizontal Scaling (Adding More Servers)

- Example: If 10,000 users visit your website, AWS adds more small servers.

2. Vertical Scaling (Increasing Server Power)

- Example: Instead of adding new servers, AWS increases the RAM and CPU power of existing servers.

3. Reactive Scaling (Responding to Demand)

- Example: If CPU usage exceeds 80%, AWS adds more servers automatically.

4. Target Tracking Scaling (Maintaining Performance)

- Example: AWS keeps CPU usage below **50%** by adjusting the number of servers dynamically.

Limitations of Auto Scaling

1. **Maximum Limit** – Each Auto Scaling group supports up to **500 instances**.
 2. **Delay in Scaling** – Adding new instances may take a few minutes.
 3. **Complex Setup** – Configuring scaling policies can be tricky.
 4. **Dependency Issues** – If databases or other services don't scale properly, your app may still slow down.
 5. **Cost Considerations** – Auto Scaling can increase costs if not configured correctly.
-

Elastic IP Address

Think of an **Elastic IP (EIP)** as a **fixed phone number** that stays with you, no matter which phone (computer/server) you are using.

Why Use an Elastic IP?

1. **Doesn't Change Over Time** → Like a personal mobile number that always stays the same.
2. **For Your AWS Account Only** → No one else can use it.
3. **Can Move Between Servers** → If one server fails, just attach it to another.
4. **Can Make Websites/Public Services Available** → Websites need a **public IP** to be accessed by users.
5. **Avoids Downtime** → If a server fails, you don't need to wait for a new IP.

How Does It Work?

Step 1: Allocate (Get) an Elastic IP

AWS gives you an Elastic IP from their **public IP pool**.

Step 2: Associate It With an Instance

Attach this Elastic IP to your **running server (EC2 instance)**.

Step 3: Use It

Now, users or customers can **access your server using this Elastic IP**.

Step 4: Reassign If Needed

If your server crashes, just move the **same Elastic IP** to another server.

Key Features

- **Elastic IP is Static** → It does not change unless you release it.
- **One Region Only** → You cannot move it to another AWS region.
- **Can Be Moved Between Servers** → If one server fails, move the IP to another.
- **Charges Apply If Not in Use** → AWS charges you if you **reserve** an Elastic IP but **do not attach** it to an instance.
- **Used in Virtual Private Cloud (VPC)** → Works inside AWS networking.
- **Can Work With Other AWS Services** → Load balancers, CloudFront, etc.

Limitations of Elastic IP

1. **Limited to 5 per AWS region** → Because public IPv4 addresses are **scarce**.
2. **Not Free If Unused** → If it's not attached to an instance, AWS **charges** for it.
3. **Region-Locked** → You cannot use an Elastic IP from **one region** in **another**.
4. **IPv4 Only** → No support for IPv6.

- **Elastic IP = A fixed public IP that you can move between servers.**

- **Used to avoid downtime, ensure reliability, and provide easy remote access.**
 - **Must be attached to a running instance to avoid charges.**
 - **Region-specific and IPv4 only.**
-

VPC

A **Virtual Private Cloud (VPC)** in AWS is like having your own **private space in the cloud**, where you can create and manage resources like servers (EC2), databases (RDS), and functions (Lambda). It gives you full control over **who can access it and how data flows** in and out.

- A **Subnet** is like dividing your VPC into smaller sections.
- **Public Subnet** → Resources here can be accessed from the internet (e.g., web servers).
- **Private Subnet** → Resources here are **not exposed to the internet** (e.g., databases).
- By default, a **VPC is private**, meaning no one from the internet can access it. To allow public access, we need an **Internet Gateway (IGW)**.
- A **Route Table** is like setting up **traffic rules** for your network.
- A **subnet** is a segment within a VPC, defining smaller network portions that help organize resources across multiple **Availability Zones (AZs)** for high availability.
- An **Internet Gateway (IGW)** is a VPC component that allows resources in the **public subnet** to communicate with the **internet**.
- A **Route Table** defines how network traffic flows within a VPC. It contains rules (routes) that determine where traffic should be directed.
- A **Security Group (SG)** acts as a **virtual firewall**, controlling inbound and outbound traffic to EC2 instances within the VPC.
- A **NAT Gateway (Network Address Translation Gateway)** allows resources in the **Private Subnet** to **access the internet** (for updates or external API calls) **without exposing them directly**.

- VPC is isolated network for AWS resources.
-

Amazon Route 53 (DNS Service)

Route 53 is a **DNS service** that translates human-readable domain names into machine-readable IP addresses.

Amazon RedShift?

Amazon Redshift is a cloud-based **data warehouse** service designed to store and analyze large amounts of data efficiently. It is fully managed, meaning AWS takes care of maintenance, scaling, and backups for you.

Example: Imagine a company that sells smartphones across different countries. They have millions of sales records and need to analyze them to see which country sells the most phones. Instead of storing all this data in a traditional database, they use Redshift to store, process, and analyze this massive dataset quickly.

- Cost effective : Starts at \$0.25 per hour, scaling up to petabyte-level data storage.
- Fast Performance
- Scalable : Can handle small datasets as well as massive petabyte-scale data.
- Managed service: AWS handles backups, security, and scaling automatically.

Redshift is an **OLAP system**, meaning it is designed for analytical queries rather than transactional operations.

Redshift consists of two types of configurations:

1. Single Node:

- Stores up to **160 GB** of data.
- Used for small datasets and testing.

2. Multi-Node:

- Used when the data is too large for a single node.
- Has two types of nodes:
 - **Leader Node:** Manages client connections, receives queries, plans execution, and coordinates with compute nodes.
 - **Compute Node:** Executes the queries and processes data in parallel before sending results to the leader node.

Think of a leader node as a **restaurant manager** who takes customer orders and delegates tasks to **chefs (compute nodes)** in the kitchen. The chefs prepare different parts of the order, and the manager compiles everything before serving the customer.

No upfront costs; pay only for what you use.

Budget And Cost Management

Budget and cost management in AWS is about **planning, monitoring, and controlling how much you're spending** on cloud services.

1. Key Concepts:

- **AWS Budgets:** Lets you set custom budgets for cost, usage, RI (Reserved Instance) utilization, and savings plans. It alerts you when your usage exceeds the set limit.
- **AWS Cost Explorer:** Helps visualize spending patterns over time.
- **Resource Planning:** Making sure you only use what you need, avoiding unnecessary costs by scaling up/down.

Imagine you're hosting an app:

- You set a budget of **₹5,000/month**.
- Using **AWS Budgets**, you get alerts when you've spent **₹4,000**.

- You use **Cost Explorer** to see which service (like EC2 or S3) is costing you the most.
- You scale down unused EC2 instances (resource planning).

Resource Planning : It's the process of estimating **what AWS resources** (like EC2, S3, RDS) your project needs to **run efficiently and cost-effectively**.

ML Ecosystem

The **ML ecosystem on AWS** includes **tools and services** that help developers build, train, deploy, and scale machine learning models without managing all the infrastructure manually.

Phases in ML Lifecycle:

1. **Data Collection & Preprocessing** – Textract, Transcribe
 2. **Model Building & Training** – SageMaker
 3. **Model Inference/Deployment** – SageMaker Endpoint, Lambda
 4. **Monitoring & Improvement** – CloudWatch, SageMaker Monitor
-

NLP

It's a **Natural Language Processing (NLP)** service that can understand and analyze text.

Example:

You run a customer feedback form. Comprehend analyzes comments like:

| "I love this product, but delivery was late."

Amazon Forecast

is a **time series forecasting service**. It predicts future values based on past data using ML.

Example:

You run a clothing store and upload sales data from the past 2 years. Forecast predicts:

| "Next month's t-shirt sales = 1200 units"

Amazon Fraud Detector

This service **uses ML to detect online fraud** such as fake accounts, payment fraud, or identity theft.

Amazon Kendra

Amazon Kendra is an **intelligent search service**. It uses **AI to answer questions** based on your documents and files.

Example:

You upload 100 company manuals. A user searches:

| "How do I reset my password?"

Kendra pulls the exact sentence from the manual where that process is explained.

Amazon Lex

It's the engine behind **chatbots**. Lex lets you **build conversational interfaces** using voice and text (like Alexa).

Example:

You build a chatbot for a restaurant:

User: "I want to book a table for 2"

Lex understands the intent = **Booking**

Amazon SageMaker

SageMaker is a **complete machine learning platform**. It helps you build, train, and deploy ML models **without needing to manage servers manually**.

Example:

You upload a dataset with house sizes and prices. SageMaker trains a model to predict:

"What would a 1500 sq ft house cost?"

Amazon Textract

Textract **automatically extracts text and data from scanned documents or images**.

Example:

You scan a form like:

Name: Rahul
DOB: 01/01/2000
Amount: ₹1500

Textract extracts the values into structured fields:

```
{"Name": "Rahul", "DOB": "01/01/2000", "Amount": "₹1500"}
```

Amazon Transcribe

Amazon Transcribe **converts speech to text** (audio → written words).

Amazon Translate

Amazon Translate **automatically translates text from one language to another** using deep learning.

Data Analytics

Data Analytics is the process of collecting, preparing, analyzing, and visualizing large sets of data to make informed decisions.

AWS provides several cloud-based services that help with different stages of the data analytics workflow: from collecting raw data, cleaning it, processing it, storing it, querying it, to visualizing it.

1. Amazon Athena – “Ask questions to your data using SQL”

Amazon Athena is a **serverless query service** that lets you run **SQL queries directly on data stored in Amazon S3**, without needing to move or load it.

Example: You have thousands of `.csv` log files in S3 and want to analyze login failures. Just write a SQL query in Athena and get the result instantly.

AWS Glue is a **serverless ETL service**. It helps you **clean, format, and move data** from different sources into a data warehouse or data lake.

Glue is an **ETL (Extract, Transform, Load)** tool:

- **Extract** = Get data from source (S3, MySQL, etc.)
- **Transform** = Clean it (e.g., fix dates, remove nulls)
- **Load** = Send it to another place (like Redshift)

Example:

You have data in **MySQL**, logs in **S3**, and **APIs from a CRM** — Glue helps merge and clean it all before sending it to **Redshift** or **Athena**.

Amazon QuickSight – “Make charts & dashboards”

QuickSight is a **Business Intelligence tool**. It helps you **make visual dashboards and graphs**.

Connects to Redshift, Athena, or S3 and shows you things like:

- Bar charts for monthly sales
- Pie charts for product category breakdowns

Amazon EMR (Elastic MapReduce) – “Process huge data using Hadoop or Spark”

EMR lets you run **big data frameworks like Hadoop and Spark** easily in the cloud. It's like a super calculator for large data.

AWS Data Pipeline – “Automate data tasks like a schedule”

What it is:

It's a **workflow tool** that lets you move or process data **on a schedule** (like every night at 12 AM).

Moves data from source → cleans it → sends to destination.

Every night, it:

- Gets data from RDS
- Uses Glue to clean it
- Loads to Redshift

AWS Lake Formation

Helps you **quickly build a secure data lake** — a central place to store all data (structured + unstructured).

You store data from:

- Excel files
 - Images
 - Logs
- ... all in one place (S3) using Lake Formation and manage access securely.

- Centralized data storage
- Access control for teams

Amazon Kinesis

Kinesis helps **collect and process data in real time**, as it happens.

A food delivery app uses Kinesis to:

- Track orders in real-time
- Alert when a delivery is late

Amazon Timestream

Timestream is for **data with time attached**, like:

- CPU usage every second
- Temperature every minute

Stores and analyzes time-stamped data efficiently.

Amazon SageMaker

SageMaker is a **machine learning platform**. It lets you **train, build, and use ML models** without needing to set up servers.

You use SageMaker to:

- Train a model to predict customer churn
- Use it in your app to warn when a customer might leave

Predict customer churn or sales

AWS Developer Tools

These tools help developers build, test, and deploy applications faster, just like a **software factory** that automates coding, testing, and shipping.

AWS X-Ray is a **distributed tracing service**. It helps you **see the flow of your app** and **find problems like delays, errors, or slow services**.

Example: You're building an e-commerce website. Customers complain that **checkout is slow**. You use X-Ray to trace where the delay is — maybe it's in the payment API.

Use AWS X-Ray to trace and analyze latency.

AWS Cloud9 is an **online IDE (Integrated Development Environment)**. You can write, run, and debug code in your browser.

Example: Instead of setting up Node.js and Python on your laptop, use Cloud9 directly in your browser and code with everything pre-installed.

Use **AWS Cloud9** to code in the browser with shared access.

AWS CodeBuild is a **build service** that compiles your code, runs tests, and produces ready-to-deploy packages.

You push your app's code to GitHub. CodeBuild automatically:

- Compiles the code
- Runs unit tests
- Gives a build artifact (like a `.zip` file)

Use **CodeBuild** in a pipeline to run builds/tests.

AWS CodeCommit is a **Git-based source control service** where developers can **store and version their code securely**.

Instead of using GitHub or Bitbucket, you host your repo in AWS CodeCommit and collaborate securely within your AWS account.

AWS CodePipeline is a **CI/CD (Continuous Integration / Continuous Delivery)** service. It helps you **automate steps from code → build → test → deploy**.

AWS Migration tools

These help you **move data, apps, and servers** from your own data center or other cloud to AWS.

AWS Migration Hub : *Central dashboard to track all migrations*

AWS Migration Hub gives a central place to track the status of multiple migration projects across tools and regions.

AWS DMS - Move databases to AWS

(Database Migration Service)DMS is a service that **migrates your database (like MySQL, Oracle) to AWS**, either one-time or ongoing with **live replication**.

AWS SMS automates the **migration of your physical/virtual servers** (e.g., from VMware, Hyper-V) to **EC2 instances** in AWS.

How to move a VM server to EC2?

Use **AWS SMS**.

Category	Tool	Purpose	Real-Life Use Case
Developer Tool	AWS X-Ray	Trace/debug apps	Finding API delays in checkout process
Developer Tool	AWS Cloud9	Online coding environment	Coding on AWS from browser
Developer Tool	AWS CodeBuild	Build and test code	Compile Java app and run tests
Developer Tool	AWS CodeCommit	Store source code (Git)	Save app code with version control

Developer Tool	AWS CodePipeline	Automate build-test-deploy	CI/CD flow for web app
Migration Tool	AWS Migration Hub	Track all migrations	Dashboard for 50-app migration
Migration Tool	AWS DMS	Migrate databases	Move SQL Server DB to Amazon RDS
Migration Tool	AWS SMS	Migrate VMs/servers	Move Windows VM to AWS EC2

