DevOps

[AWS Services Overview 2](#_Toc193568690)

[AWS Overview and Shared Responsibility Model 2](#_Toc193568691)

[Elasticity 3](#_Toc193568692)

[High Availability (HA) 4](#_Toc193568693)

[AWS Global Infrastructure - Regions and Availability Zones (AZs) 6](#_Toc193568694)

[EC2, Load Balancer, Elastic Beanstalk & LightSail 8](#_Toc193568695)

[AWS Lambda for EC2 Automation 10](#_Toc193568696)

[S3 (Simple Storage Service) 12](#_Toc193568697)

[Elastic Block Storage (EBS) 14](#_Toc193568698)

[AWS Storage Services 16](#_Toc193568699)

[Database Services 18](#_Toc193568700)

[AWS ElastiCache 19](#_Toc193568701)

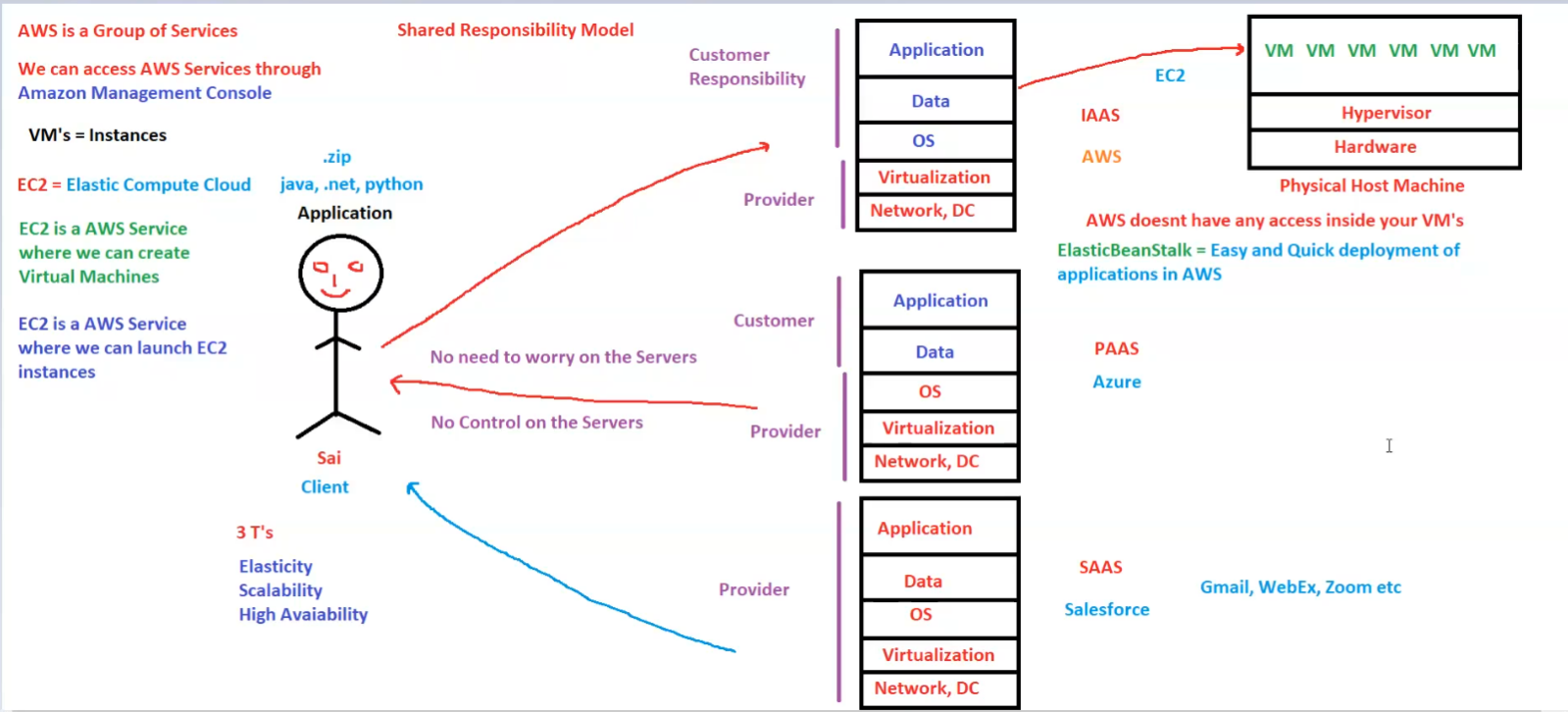
[Networking Services 21](#_Toc193568702)

[CloudFront 22](#_Toc193568703)

[AWS Monitoring, Security, and Support Overview 24](#_Toc193568704)

# AWS Services Overview

## AWS Overview and Shared Responsibility Model

****

Here are the notes based on the image:

**AWS Overview and Shared Responsibility Model**

**1. AWS Basics**

* **AWS (Amazon Web Services)** is a group of cloud services.
* We can access AWS services via the **Amazon Management Console**.
* **VMs (Virtual Machines) = Instances** in AWS.

**2. EC2 (Elastic Compute Cloud)**

* **EC2** is an AWS service that allows users to create and launch virtual machines.
* **EC2 falls under the Infrastructure as a Service (IAAS) model**.
* AWS does not have access inside your virtual machines.

**3. Shared Responsibility Model**

* **Customer Responsibility**: Application and Data management.
* **Provider Responsibility**: Network, Data Center, and Virtualization.
* **Shared Responsibility**:
  + Customers do not need to worry about servers.
  + Customers have no control over the physical servers.

**4. Cloud Service Models**

1. **IAAS (Infrastructure as a Service)**
   * Example: **AWS EC2**
   * Customers manage applications, data, and OS, while AWS handles virtualization and infrastructure.
2. **PAAS (Platform as a Service)**
   * Example: **Azure, AWS Elastic Beanstalk**
   * Provides an environment for application development and deployment.
3. **SAAS (Software as a Service)**
   * Examples: **Salesforce, Gmail, WebEx, Zoom**
   * Fully managed applications accessible via the internet.

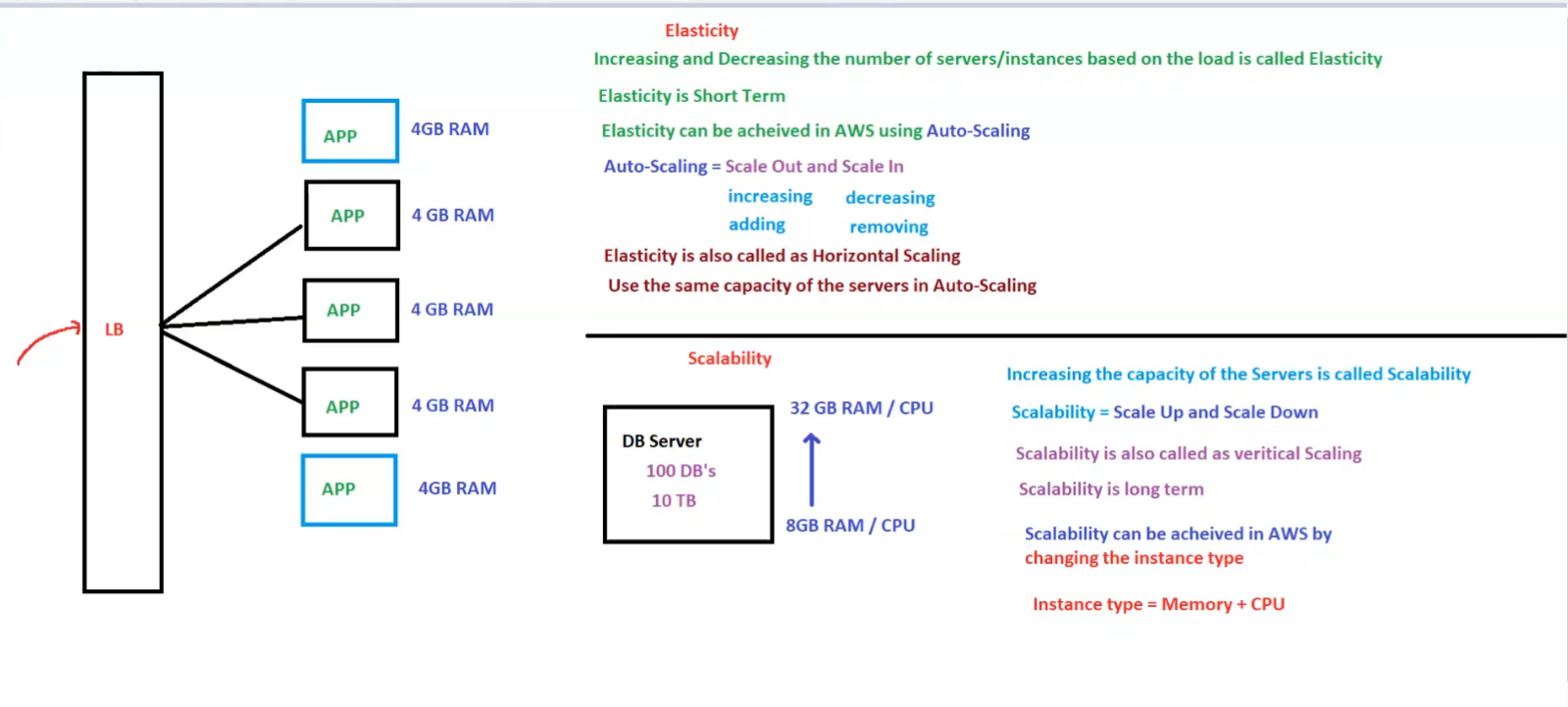
**5. Benefits of AWS (The 3 T’s)**

* **Elasticity**
* **Scalability**
* **High Availability**

**6. AWS Infrastructure**

* AWS physical host machines run multiple virtual machines (VMs) using a **Hypervisor**.
* **Elastic Beanstalk** is an AWS service for easy and quick application deployment.

## Elasticity



**Elasticity**

* **Definition**: Increasing and decreasing the number of servers/instances based on the load is called **Elasticity**.
* **Characteristics**:
  + Elasticity is **short-term**.
  + Achieved in **AWS using Auto-Scaling**.
  + **Auto-Scaling** involves:
    - **Scale Out** (increasing/adding instances)
    - **Scale In** (decreasing/removing instances)
  + Also known as **Horizontal Scaling**.
  + Uses **same capacity servers** in Auto-Scaling.

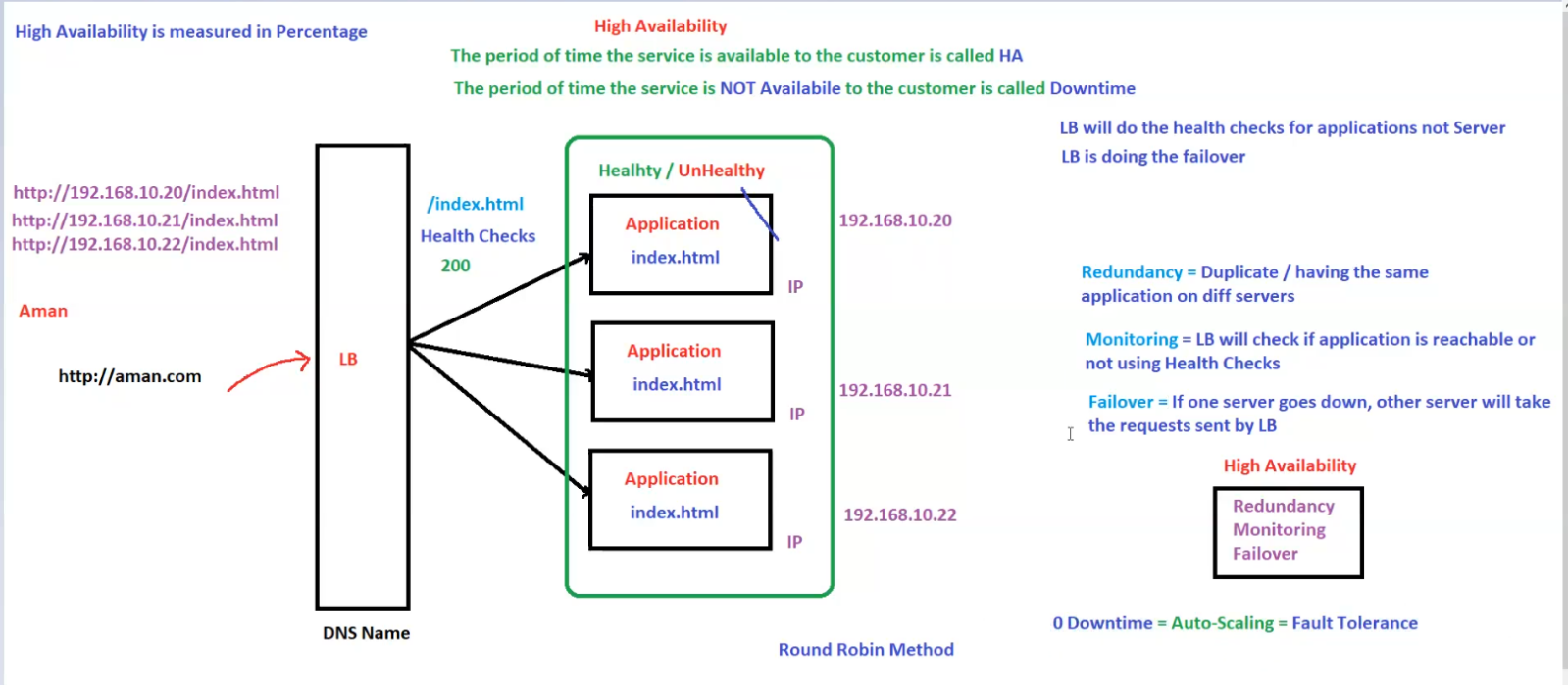
**Scalability**

* **Definition**: Increasing the **capacity** of servers is called **Scalability**.
* **Characteristics**:
  + Scalability = **Scale Up and Scale Down**.
  + Also called **Vertical Scaling**.
  + Scalability is **long-term**.
  + Achieved in **AWS by changing the instance type**.
  + **Instance Type** = **Memory + CPU**.

**Key Differences**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Elasticity (Horizontal Scaling)** | **Scalability (Vertical Scaling)** |
| **Definition** | Adjusting the number of instances | Increasing server capacity |
| **Timeframe** | Short-term | Long-term |
| **AWS Feature** | Auto-Scaling | Changing Instance Type |
| **Process** | Scale In/Out (adding/removing instances) | Scale Up/Down (increasing/decreasing resources) |
| **Capacity** | Uses same capacity servers | Increases capacity of the same server |

## High Availability (HA)

****

**High Availability (HA)**

* **Definition**: The period of time the service is **available** to the customer is called **High Availability (HA)**.
* **Downtime**: The period of time the service is **not available** to the customer is called **Downtime**.
* **HA is measured in percentage**.

**Key Components of High Availability**

1. **Redundancy**:
   * **Definition**: Having duplicate instances of the same application on different servers.
2. **Monitoring**:
   * **Definition**: Load Balancer (LB) checks if the application is reachable using **Health Checks**.
   * Health Check: Verifies the availability of **index.html** and expects a **200 status**.
3. **Failover**:
   * **Definition**: If one server goes down, other servers take over the requests handled by the Load Balancer.

**Load Balancer (LB)**

* **Handles incoming traffic and distributes it across multiple servers**.
* **Performs Health Checks** for applications (not servers).
* **Performs failover** to redirect traffic if a server goes down.
* **Uses Round Robin Method** for load distribution.

**Ensuring 0 Downtime**

* **Auto-Scaling = Fault Tolerance**
* **Combining**:
  + **Redundancy**
  + **Monitoring**
  + **Failover**  
    Leads to **High Availability**.

**Example Setup**

* **DNS Name:** http://aman.com
* **Backend servers with IPs**:
  + 192.168.10.20
  + 192.168.10.21
  + 192.168.10.22
* Each server hosts index.html, and the Load Balancer distributes traffic among them.

## AWS Global Infrastructure - Regions and Availability Zones (AZs)

**AWS Global Infrastructure - Regions and Availability Zones (AZs)**

**1. AWS Global Infrastructure Overview**

* **Region**: A physical location where AWS has infrastructure (e.g., Mumbai).
* **Availability Zone (AZ)**: A **data center** within a region.
* **A Region has multiple AZs**, and each AZ is a group of **data centers** (DCs).
* **Servers (EC2 Instances)** are placed within **AZs**.

**2. AWS Region and Availability Zones**

* **Example**: AWS Region **Mumbai** (ap-south-1) has **three AZs**:
  + **ap-south-1a**
  + **ap-south-1b**
  + **ap-south-1c**
* **AZs are managed by AWS** and are **interconnected via a network**.
* **Best Practice**: Distribute instances across **multiple AZs** for high availability.

**3. Communication Between AZs & Regions**

* **AZs within a region** can communicate with each other **by default**.
* **Regions do NOT communicate by default**, but it can be enabled if required.
* **Low Latency (1 sec) is good**, while **high latency (5 sec) is bad**.

**4. Load Balancing & Traffic Distribution**

* **Load Balancer (LB)**:
  + Distributes traffic across **multiple EC2 instances** in different **AZs**.
  + **LB is specific to a region**, not an AZ.
* **Instances across AZs** can share data if required.

**5. High Availability & Fault Tolerance**

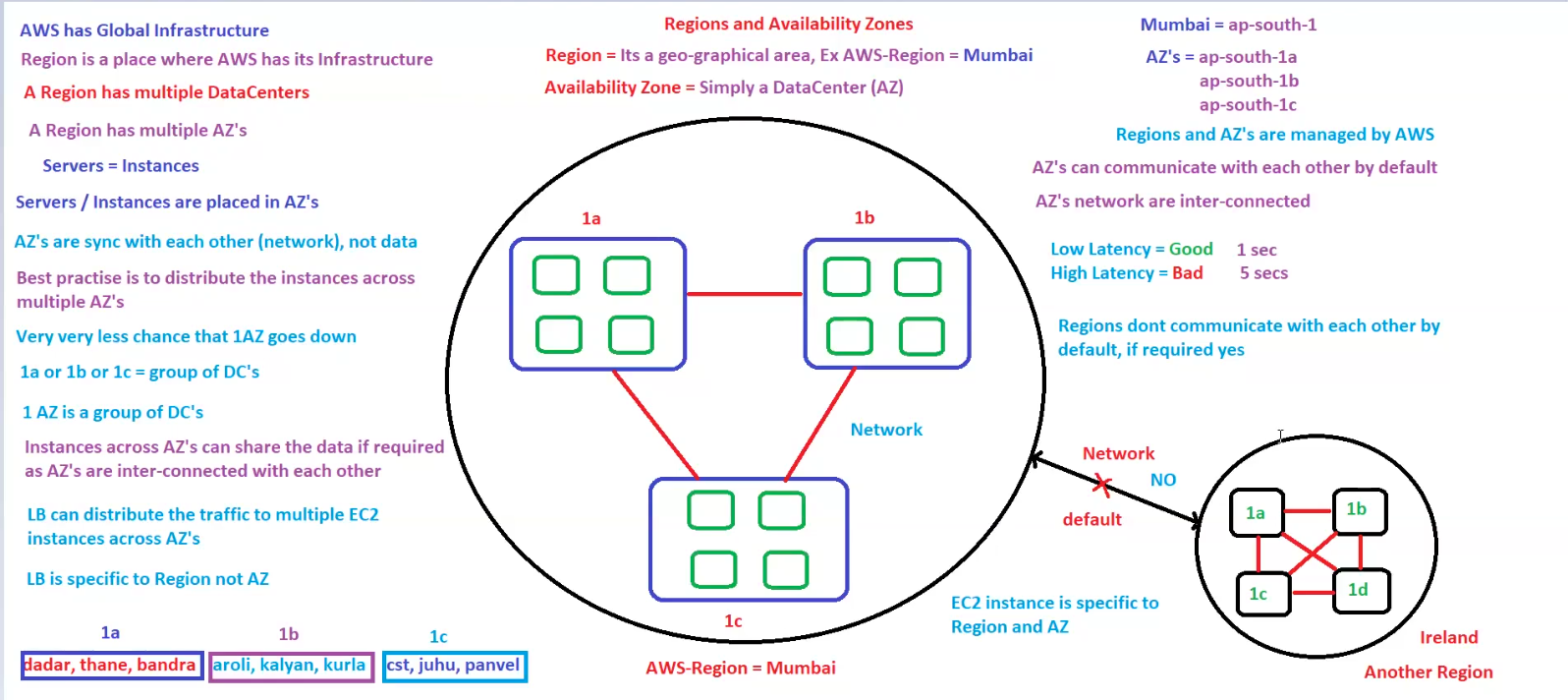
* **Very low chance** that an entire **AZ goes down**.
* **Distributing instances across AZs** ensures **fault tolerance**.
* **If one AZ fails, others continue running**.

**6. Example Mapping (Mumbai Region)**

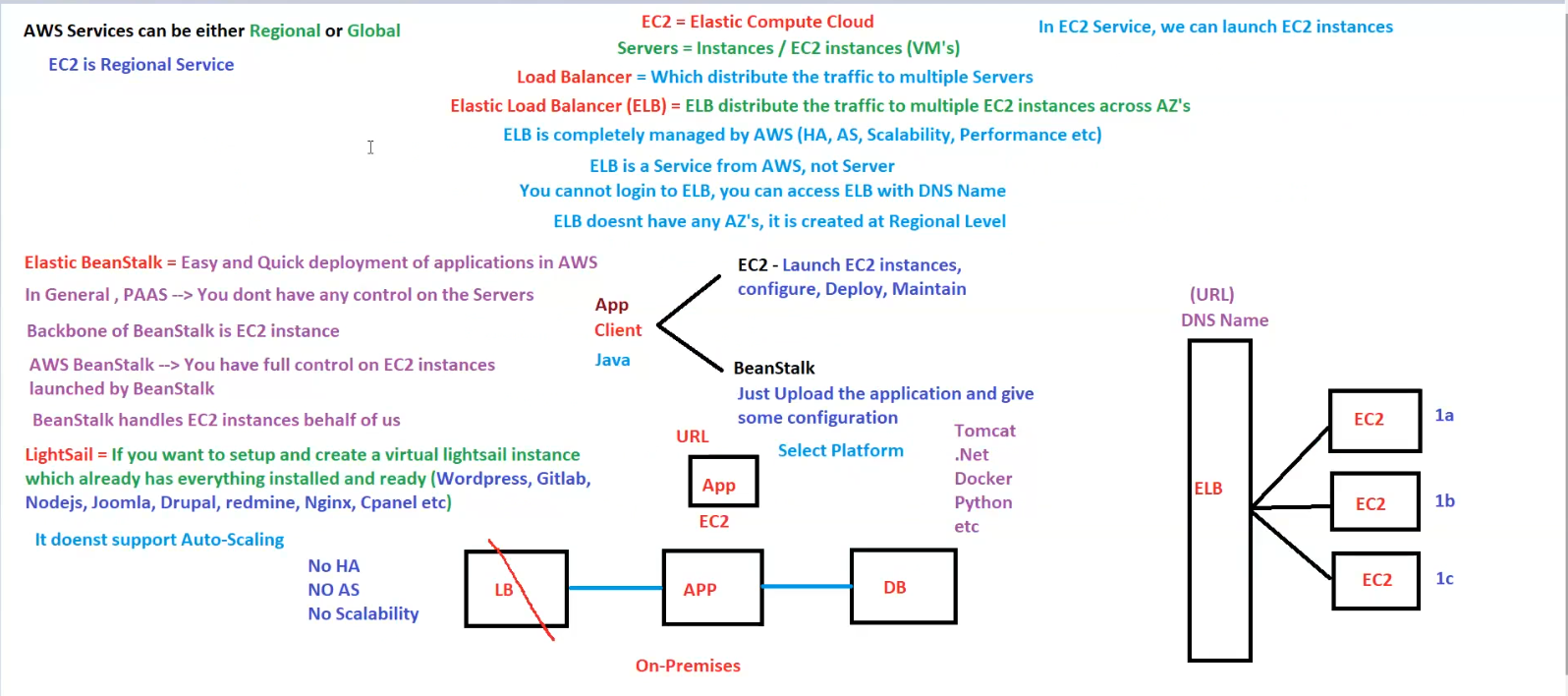
* **ap-south-1a**: Dadar, Thane, Bandra
* **ap-south-1b**: Airoli, Kalyan, Kurla
* **ap-south-1c**: CST, Juhu, Panvel

**7. Summary**

✅ **Region** = Group of AZs.  
✅ **AZ** = Group of Data Centers (DCs).  
✅ **AZs within a region can communicate**.  
✅ **Regions do NOT communicate by default**.  
✅ **Load Balancers distribute traffic across AZs**.  
✅ **Best practice**: Use multiple AZs for **high availability & redundancy**.



## EC2, Load Balancer, Elastic Beanstalk & LightSail



**1. AWS Services - Regional vs Global**

* **AWS Services can be either Regional or Global.**
* **EC2 (Elastic Compute Cloud) is a Regional Service**
  + It means EC2 instances exist within a **specific AWS region**.
  + EC2 instances are virtual machines (VMs) in the cloud.

**2. EC2 Instances**

* **EC2 = Elastic Compute Cloud** → Allows launching virtual servers (instances).
* **Servers = Instances (EC2 VMs).**
* EC2 instances need to be **configured, deployed, and maintained manually**.

**3. Load Balancer (LB)**

* **Distributes incoming traffic** across multiple EC2 instances.
* **Elastic Load Balancer (ELB)**
  + **ELB distributes traffic across EC2 instances in different AZs.**
  + **Managed by AWS** (High Availability, Auto Scaling, Performance Optimization).
  + **Created at the Regional Level**, not tied to a specific AZ.
  + **Cannot log in to ELB**, only access it via DNS name.

**4. Elastic Beanstalk (PaaS)**

* **Simplifies application deployment** on AWS.
* **Fully managed service**, handling EC2 instances, Load Balancing, and Auto Scaling.
* **Key Features:**
  + Uses **EC2 instances as the backbone**.
  + You have **full control over EC2 instances**.
  + Just **upload your application** and provide configuration.
  + Supports **multiple platforms** like **Tomcat, .Net, Docker, Python, etc.**

**5. AWS LightSail (Simplified VM Hosting)**

* **Pre-configured cloud service** for hosting applications quickly.
* Supports **WordPress, GitLab, Node.js, Joomla, Drupal, Nginx, Cpanel, etc.**
* **Limitations**:
  + ❌ **No Auto-Scaling**.
  + ❌ **No High Availability (HA)**.
  + ❌ **No Scalability** (Cannot dynamically scale like EC2 with ELB).

**6. EC2 Deployment Options**

**A. EC2 with ELB (Recommended)**

✅ EC2 instances are deployed across **multiple AZs**.  
✅ **ELB distributes traffic** to maintain high availability.  
✅ **Auto Scaling** ensures instances scale up/down as needed.

**B. EC2 with Beanstalk**

✅ Easy deployment and management.  
✅ **AWS manages scaling & load balancing**.  
✅ **Ideal for developers** who don’t want to manage infrastructure.

**C. EC2 Without Load Balancer (Not Recommended)**

❌ Single EC2 instance without redundancy.  
❌ **No auto-scaling** = No ability to handle high traffic.  
❌ **If the instance fails, the application is down.**

**7. Summary**

| **AWS Service** | **Use Case** | **Key Features** |
| --- | --- | --- |
| **EC2** | General-purpose compute | Manual setup, requires maintenance |
| **ELB** | Load balancing traffic | Auto-distributes requests across EC2 instances |
| **Elastic Beanstalk** | Easy application deployment | PaaS, fully managed, supports multiple platforms |
| **LightSail** | Quick app hosting | Simple VM setup, no scaling, no HA |

✅ **Use EC2 + ELB for scalable and fault-tolerant architecture.**  
✅ **Use Beanstalk for easy deployment with AWS-managed scaling.**  
✅ **Use LightSail for simple, pre-configured environments without scaling needs.**

## AWS Lambda for EC2 Automation

**1. AWS Lambda Overview**

* **Lambda is a Serverless service** → No need to manage servers.
* You can **run code** without provisioning infrastructure.
* Supports **multiple languages** like **Java, Python, .NET, Ruby, etc.**
* Lambda can be **triggered by events** (e.g., EC2 state changes, scheduled tasks, etc.).

**2. Automating EC2 with Lambda**

* **Lambda can perform EC2 operations**:  
  ✅ **Launch**  
  ✅ **Stop**  
  ✅ **Start**  
  ✅ **Reboot**  
  ✅ **Terminate/Delete**
* **Lambda is triggered based on:**
  + **EventBridge Rules** (For Scheduled Events).
  + **Launch Events** (For EC2 instance events).

**3. Event-Based EC2 Management Using Lambda**

**A. Scheduled Stop & Start of EC2 Instances**

* **9 PM → STOP EC2 Instances**
  + **EventBridge Rule triggers STOPFn Lambda function**.
  + Lambda stops all EC2 instances.
* **6 AM → START EC2 Instances**
  + **EventBridge Rule triggers StartFn Lambda function**.
  + Lambda starts all EC2 instances.

**B. EC2 Termination Using Lambda**

* **Lambda can terminate EC2 instances using Python Code (TerminateFn).**
* This can be invoked manually or automatically via **EventBridge**.
* Example Use Case: **Auto-terminate idle EC2 instances** to reduce costs.

**4. AWS Services Involved**

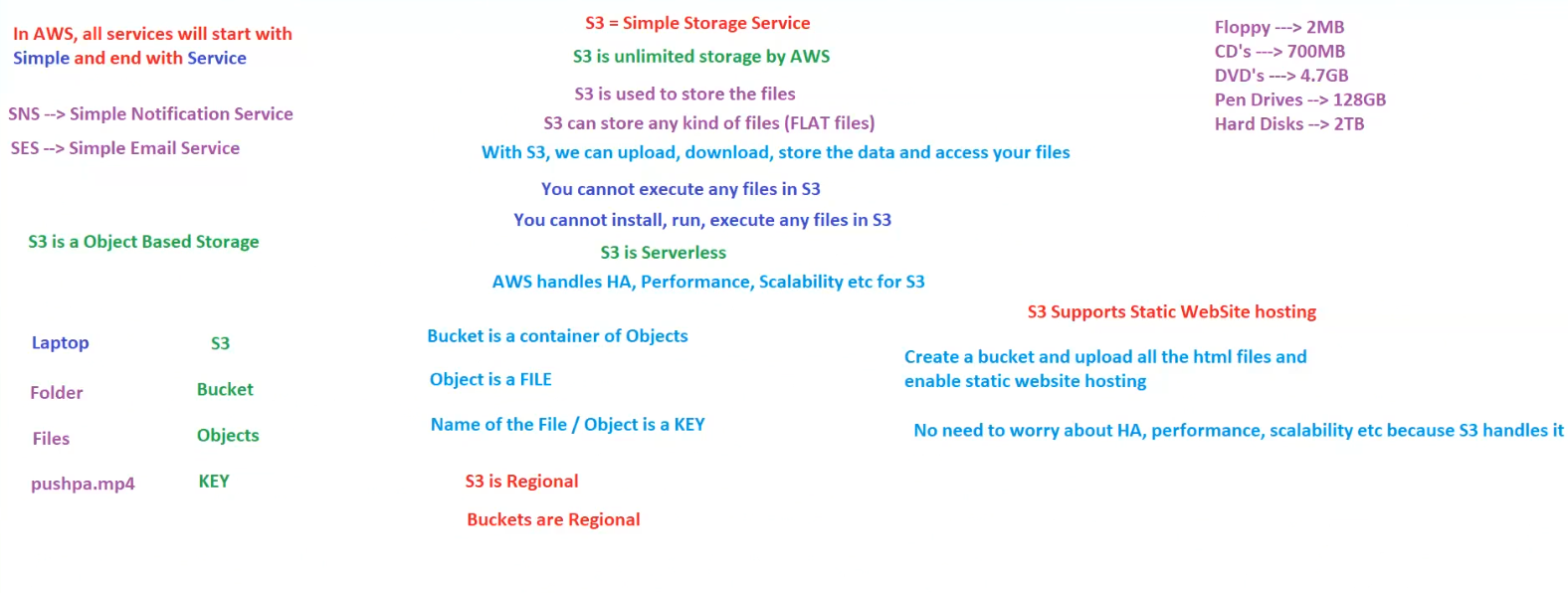
| **Service** | **Purpose** |
| --- | --- |
| **Lambda** | Runs Python code to start/stop EC2 instances. |
| **EventBridge** | Schedules events (9 PM Stop, 6 AM Start). |
| **EC2** | Instances managed via Lambda automation. |

**5. Summary**

✅ **AWS Lambda is used for automating EC2 instance operations.**  
✅ **EventBridge rules** can **trigger Lambda functions** to start/stop EC2 instances **on schedule**.  
✅ **Lambda functions** can be used for **launching, stopping, and terminating EC2 instances automatically**.  
✅ **Serverless automation** reduces manual effort & optimizes cloud costs.



## S3 (Simple Storage Service)



**1. What is AWS S3?**

* **S3 = Simple Storage Service**.
* **S3 provides unlimited storage** managed by AWS.
* **S3 is used for storing files** (object-based storage).
* **Supports any type of file** (flat files like images, videos, documents, backups, etc.).
* **S3 is Serverless** – No need to manage infrastructure.
* **AWS handles High Availability (HA), Performance, and Scalability**.

**2. How S3 Works? (Folder & File Concept)**

| **Local System (Laptop)** | **S3 Equivalent** |
| --- | --- |
| **Laptop** | **S3** |
| **Folder** | **Bucket** |
| **Files** | **Objects** |
| **File Name** | **Key** |

* **Bucket**: A container for storing objects (files).
* **Object**: A stored file.
* **Key**: The unique name assigned to an object (file name).

**3. S3 Storage Features & Limitations**

✅ **Can upload, download, and store files**.  
❌ **Cannot execute, install, or run any files directly in S3**.  
✅ **Supports Static Website Hosting** (for hosting HTML files).  
✅ **S3 is Regional** → Buckets exist within a specific AWS region.  
❌ **Buckets are also Regional** (Cannot access from different regions unless configured).

**4. S3 Static Website Hosting**

* **S3 supports static website hosting**.
* **Steps to host a website on S3**:
  1. **Create a Bucket**.
  2. **Upload HTML files**.
  3. **Enable static website hosting**.
  4. **S3 will manage HA, Performance, and Scalability**.

✅ No need for additional infrastructure like EC2.

**5. Comparison of Traditional Storage vs S3**

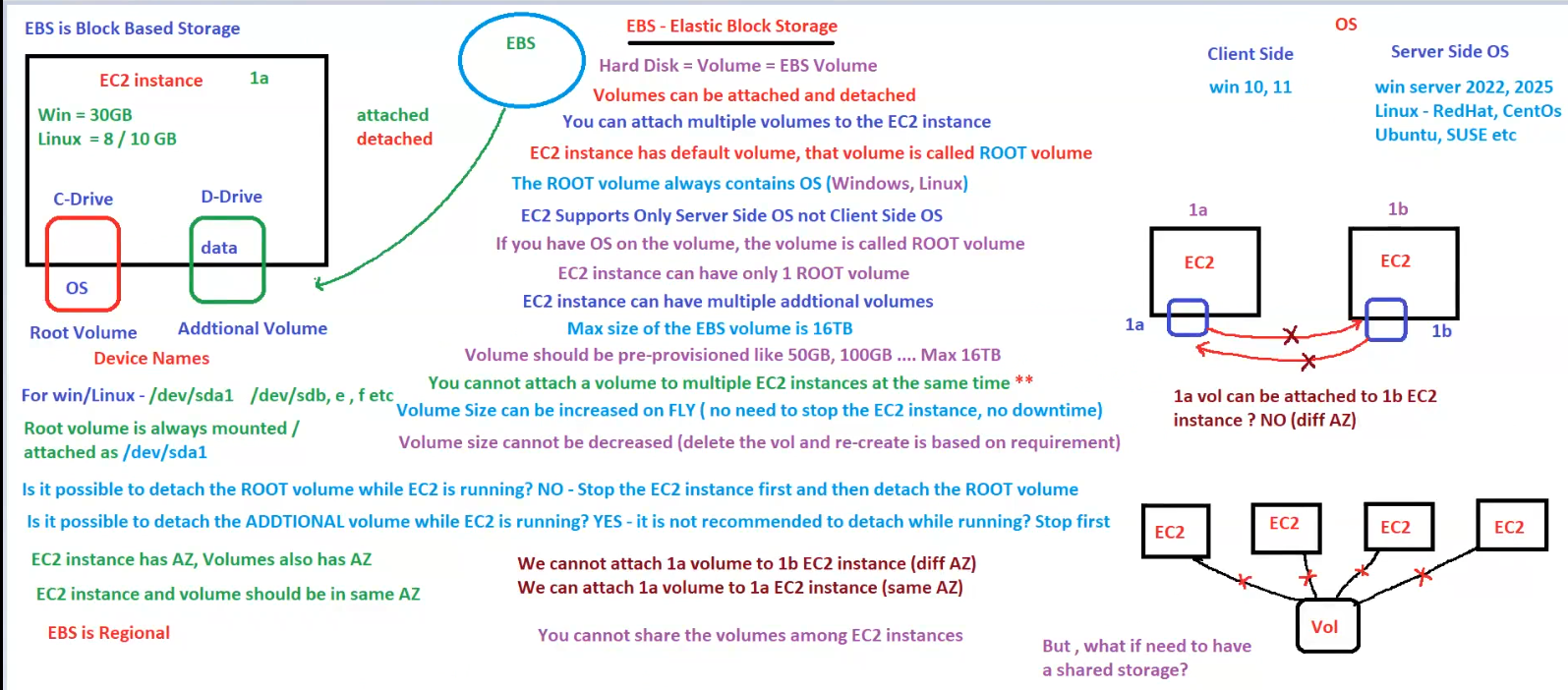
| **Storage Type** | **Capacity** |
| --- | --- |
| **Floppy Disk** | 2MB |
| **CD** | 700MB |
| **DVD** | 4.7GB |
| **Pen Drive** | 128GB |
| **Hard Disk** | 2TB |
| **S3** | **Unlimited** ✅ |

**6. Summary**

| **Feature** | **Details** |
| --- | --- |
| **Type** | Object Storage |
| **Storage Limit** | Unlimited ✅ |
| **Execution** | Cannot execute files ❌ |
| **Static Website Hosting** | Supported ✅ |
| **Region** | S3 is Regional ✅ |
| **Bucket Region** | Buckets are Regional ✅ |

✅ **Use S3 for file storage, backups, and hosting static websites**.  
✅ **For shared storage across multiple regions, enable S3 Cross-Region Replication**.

## Elastic Block Storage (EBS)



**1. What is EBS (Elastic Block Storage)?**

* **EBS is a block-based storage system** for EC2 instances.
* Acts like a **hard disk (volume)** that can be **attached and detached** from EC2 instances.
* **EBS is Regional** → Volumes exist within a specific AWS region.
* **Each volume is tied to a single Availability Zone (AZ)**.

**2. EC2 and EBS Volumes**

* Every **EC2 instance has a default volume**, called the **ROOT volume**.
* The **Root Volume always contains the OS** (Windows/Linux).
* EC2 instances **can have multiple additional volumes** for storing extra data.

**3. Volume Types in EC2**

* **Root Volume**:
  + Contains the **OS**.
  + **Always mounted at /dev/sda1** on Linux/Windows.
  + Cannot be detached while EC2 is running → **Instance must be stopped first**.
* **Additional Volume**:
  + Used for **storing data**.
  + Can be detached while EC2 is running, but **not recommended**.

**4. EBS Features & Limitations**

✅ **Supports only Server-Side OS** (Windows Server, Linux).  
✅ **Max EBS Volume Size = 16TB**.  
✅ **EBS Volume can be increased on the fly** (No need to stop the EC2 instance).  
❌ **Cannot decrease volume size** (Need to create a new volume and migrate data).  
❌ **A volume cannot be attached to multiple EC2 instances at the same time**.  
❌ **An EBS volume is limited to a single AZ** (Cannot attach to EC2 in another AZ).

**5. Attaching & Detaching Volumes**

| **Action** | **Allowed?** | **Notes** |
| --- | --- | --- |
| **Detach Root Volume while EC2 is running** | ❌ No | Stop EC2 first, then detach. |
| **Detach Additional Volume while EC2 is running** | ✅ Yes | Not recommended, may cause data corruption. |
| **Attach Volume to another EC2 in the same AZ** | ✅ Yes | Allowed if in the same AZ. |
| **Attach Volume to an EC2 in a different AZ** | ❌ No | EBS is AZ-specific. |

**6. Shared Storage Between EC2 Instances?**

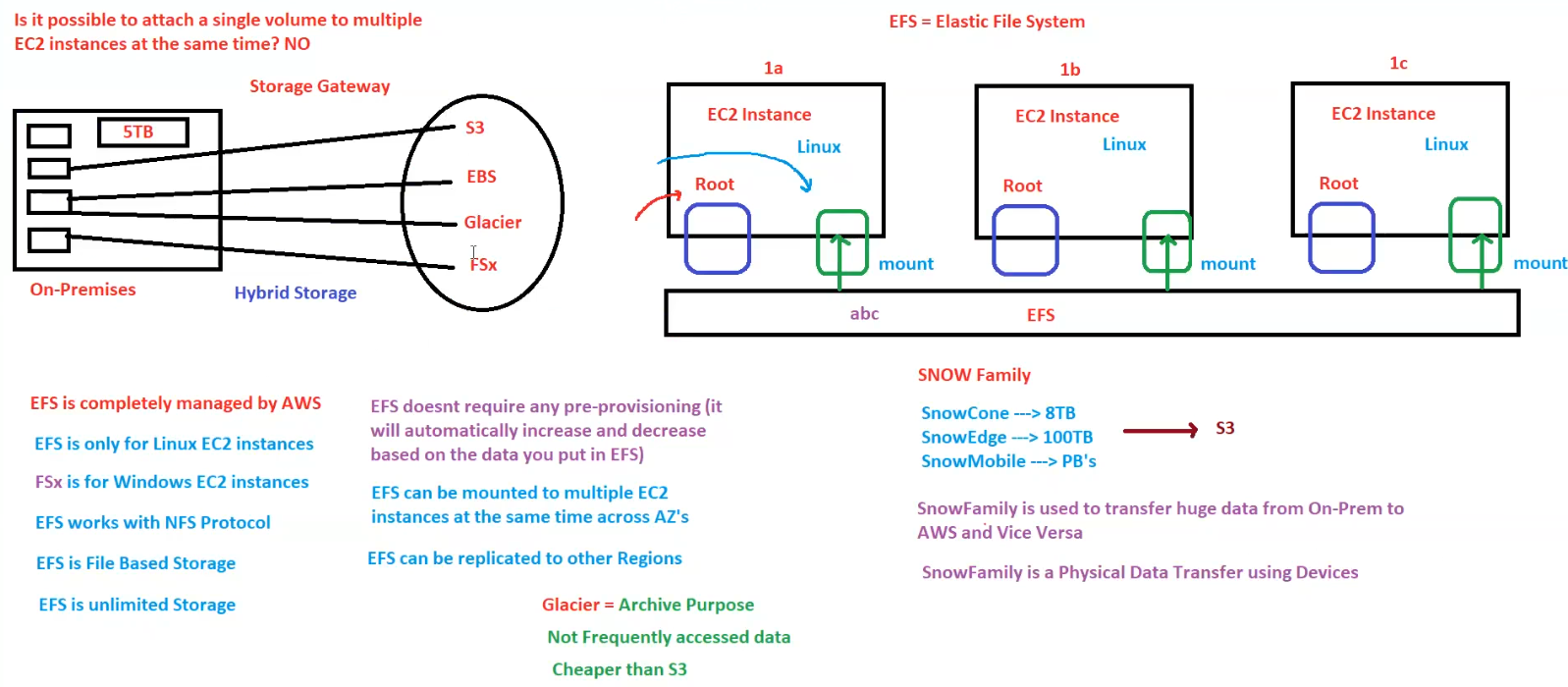
* **EBS volumes cannot be shared across multiple EC2 instances**.
* **Alternative**: Use **EFS (Elastic File System) or FSx** for shared storage.

**7. Summary**

| **Feature** | **Details** |
| --- | --- |
| **Root Volume** | OS Storage, mounted as /dev/sda1, cannot detach while running. |
| **Additional Volume** | Extra storage, can be attached/detached while running. |
| **Max Volume Size** | **16TB** |
| **Resize Volume** | ✅ Increase without downtime, ❌ Cannot decrease. |
| **Multi-Attach** | ❌ Cannot attach to multiple EC2 instances at the same time. |
| **Cross-AZ Attachment** | ❌ EBS volume is restricted to a single AZ. |

✅ **Use EBS for block storage**.  
✅ **For shared storage, use EFS or FSx**.

## AWS Storage Services



**AWS Storage Services Overview**

**1. Attaching a Single Volume to Multiple EC2 Instances**

* **Not Possible** to attach a single EBS volume to multiple EC2 instances at the same time.

**2. Hybrid Storage (On-Premises & AWS)**

* Storage Gateway enables on-premises data to connect with AWS services like:
  + **S3** (Object Storage)
  + **EBS** (Block Storage)
  + **Glacier** (Archive Storage)
  + **FSx** (Windows File System)

**3. Elastic File System (EFS)**

* **EFS = Elastic File System**
* **Fully managed** by AWS.
* **Only for Linux EC2 Instances**.
* Works with **NFS Protocol**.
* **File-based storage** solution.
* **Unlimited storage capacity**.
* **No pre-provisioning required** – it automatically scales up or down.
* **Can be mounted to multiple EC2 instances** across Availability Zones.
* **Supports replication across regions**.

**4. FSx (File System)**

* **FSx is for Windows EC2 instances**.
* **Not the same as EFS**.

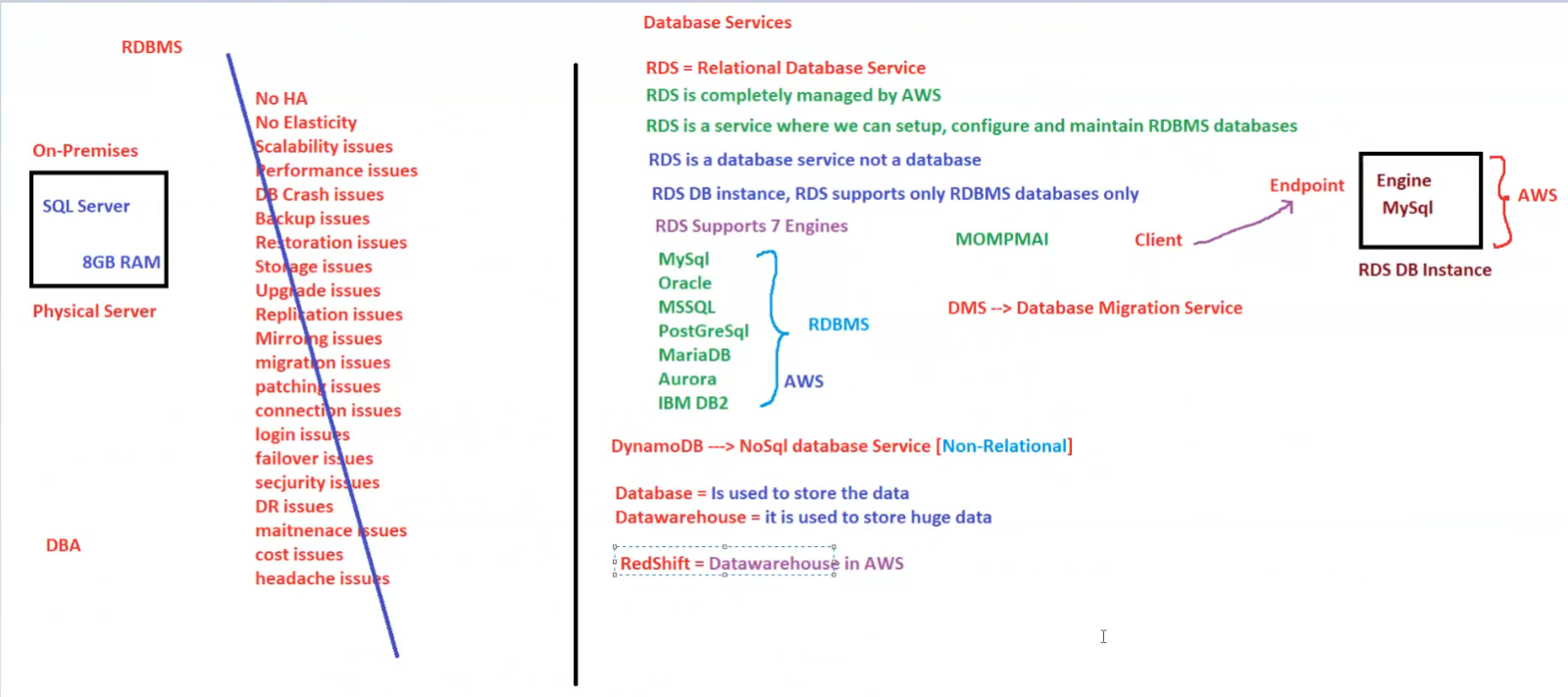
**5. Glacier**

* Used for **archival storage**.
* **Not frequently accessed data**.
* **Cheaper than S3**.

**6. AWS Snow Family (Data Transfer Solutions)**

* **SnowCone** → 8TB
* **SnowEdge** → 100TB
* **SnowMobile** → Petabytes (PBs)
* Used for **large-scale data transfer** from **on-prem to AWS and vice versa**.
* **Physical data transfer** using devices.

## Database Services



Here are the notes based on the image:

**AWS Database Services Overview**

**1. On-Premises RDBMS Challenges**

Traditional on-premises SQL Server databases face several challenges, including:

* **No high availability features** (No HA)
* **No automatic scalability**
* **Performance issues**
* **Scalability issues**
* **Database crashes**
* **Backup & restoration challenges**
* **Security risks**
* **Patching & maintenance complexities**
* **Connection and login issues**
* **Failover issues**

**2. AWS RDS (Relational Database Service)**

* **RDS (Relational Database Service)** is **completely managed by AWS**.
* Allows easy setup, configuration, and maintenance of **RDBMS databases**.
* RDS supports **7 database engines**:
  + **MySQL**
  + **PostgreSQL**
  + **MSSQL**
  + **MariaDB**
  + **Aurora (AWS native)**
  + **IBM DB2**
  + **Oracle RDS**
* **Automates**:
  + Backups
  + Software patching
  + Automatic scaling
  + High availability & failover support

**3. AWS Database Solutions**

* **DynamoDB** – NoSQL database service for high performance & scalability.
* **Amazon RDS DB Instance** (fully managed relational database in AWS).
* **DMS (Database Migration Service)** – Helps migrate databases from **on-premises to AWS**.
* **Amazon Redshift** – **AWS’s data warehousing solution**.

## AWS ElastiCache

Here are the notes based on the image:

**AWS ElastiCache - In-Memory Database Caching Service**

**1. Overview**

* **ElastiCache** is an in-memory caching service provided by AWS.
* It improves **application performance** by reducing the load on the primary database.
* Key benefits:
  + **Low latency**
  + **High performance**
  + **Faster data retrieval** for frequently accessed data.

**2. System Architecture Overview**

1. **Client (User Request)**
   * Example: User searches for "Shoes" on Flipkart.
   * Request is sent to the **Application Server**.
2. **Application Server**
   * Receives the request from the user.
   * Checks if the data is available in the **Cache Memory** before querying the **Database Server**.
3. **Cache Memory (ElastiCache)**
   * Stores **frequently accessed data**.
   * Helps reduce **database load** and improve response time.
   * Uses in-memory caching solutions like:
     + **Redis**
     + **Memcached**
4. **Database Server**
   * If data is not found in the cache, the application server queries the **database server**.
   * The database server fetches the required data from **HDD (Hard Disk Drive)** if not available in cache.
   * The response is sent back to the **application server**.
   * Frequently accessed data is stored in **cache memory** for faster retrieval in future requests.

**Key Takeaways**

* **Caching** helps reduce latency and improves performance.
* **AWS ElastiCache** supports caching with **Redis** and **Memcached**.
* Frequently accessed data is **cached to reduce direct database queries**.
* **Application Servers first check cache memory** before querying the **Database Server**.
* If data is not in cache, the database fetches it from the HDD and sends it to both the cache and the application server.
* This **reduces load on the database** and makes applications **faster & more scalable**.

## Networking Services

Here are the notes based on the image:

**AWS Networking Services Overview**

**1. Route 53 (DNS Service)**

* **Route 53 (R53) is a DNS Service from AWS**.
* Uses **DNS Port 53**.
* **Global Service** (not limited to a specific region).
* **Handles DNS records & mapping** for domain names.
* Can be used to map **user-friendly domain names** to AWS resources.
* **Example Problem:**
  + AWS ELB (Elastic Load Balancer) generates a **long and complex URL** like:  
    http://myelb.8457876.ap-south-1.elb.amazonaws.com
  + **Solution:** Use **Route 53** to map this to a custom domain, making it user-friendly.

**2. Virtual Private Cloud (VPC)**

* **VPC = Virtual Private Cloud**
* A **virtual datacenter on AWS**.
* **VPC is regional** (limited to a single AWS region).
* Each AWS region has a **default VPC**.
* **VPC contains:**
  + **Availability Zones (AZ)**
  + **EC2 instances**
  + **Subnets**
  + **Security Groups**
  + **Internet Gateways (IGW)**

**3. Connectivity Between On-Premises and AWS**

**VPN (Virtual Private Network)**

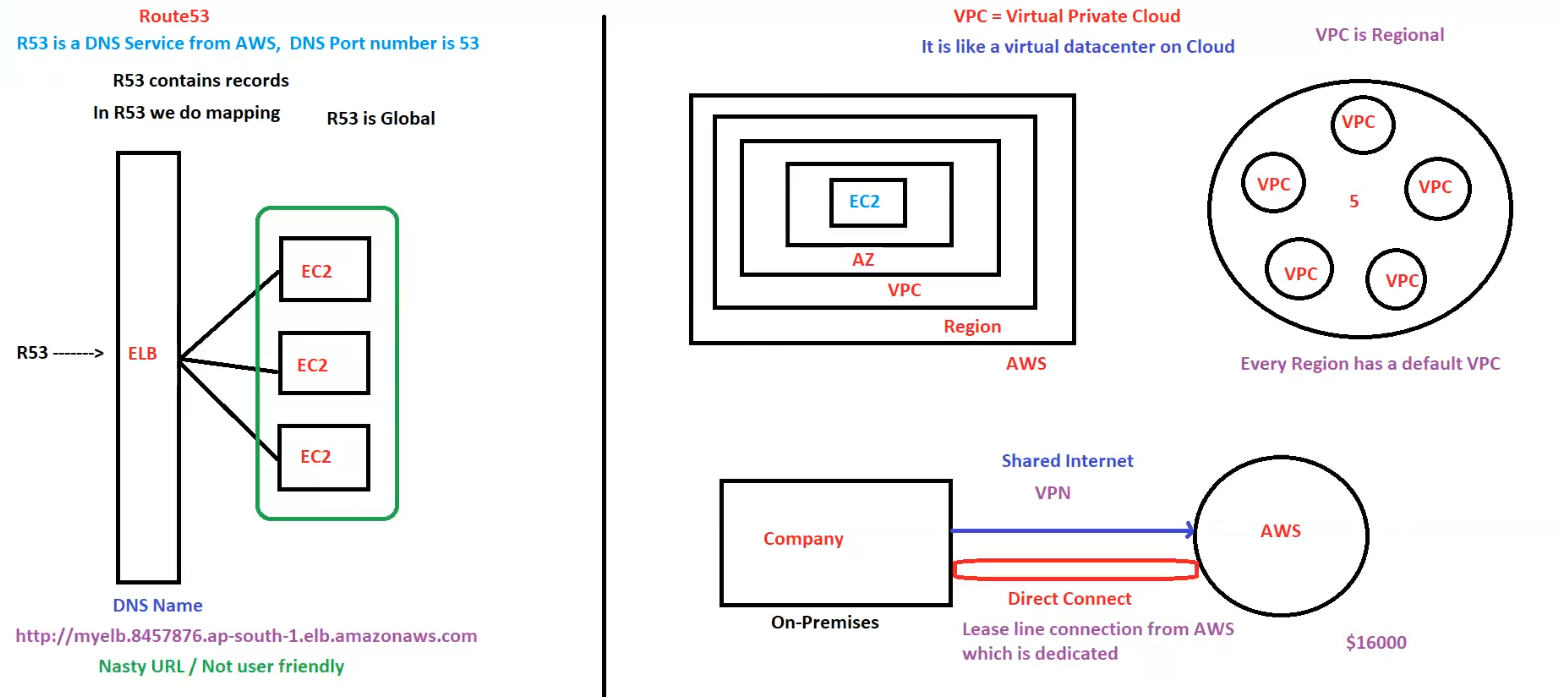
* Uses a **shared internet connection**.
* **Cheaper but less reliable** than Direct Connect.

**AWS Direct Connect**

* **Dedicated lease line** from AWS to the on-premises data center.
* **More stable & secure** than VPN.
* **Cost:** Around **$16,000** for a dedicated connection.

**Key Takeaways**

* **Route 53** helps in mapping domain names to AWS resources.
* **VPCs are regional** and act as virtual datacenters.
* **Direct Connect is a better alternative to VPN** for stable, high-speed connections between on-prem and AWS.



## CloudFront

Here are the notes based on the image:

**AWS CloudFront & Route 53 Overview**

**1. AWS Route 53 (DNS Service)**

* **Route 53 (R53) is AWS's DNS service**.
* **Port number**: 53 (standard DNS port).
* **Global service** (not restricted to a specific region).
* **Contains DNS records & does domain mapping**.
* Can be used to route traffic to AWS resources like **Elastic Load Balancer (ELB)** and **EC2 instances**.

**2. AWS CloudFront (CDN - Content Delivery Network)**

* **CloudFront is a CDN (Content Delivery Network) from AWS**.
* **Reduces latency & improves performance** by caching content at AWS **Edge Locations**.
* **Edge Locations (ELs)**:
  + Act as caching servers.
  + Store **static and dynamic data** close to users.
  + Managed by **AWS**.
* **TTL (Time to Live)**:
  + Determines how long data is cached before it is refreshed.
  + Example: **TTL = 12 hours** (after this, cache is updated).
* **Cache Invalidation**:
  + Deletes old cache before TTL expires.
  + Useful when content updates need immediate effect.

**3. Without vs. With CloudFront**

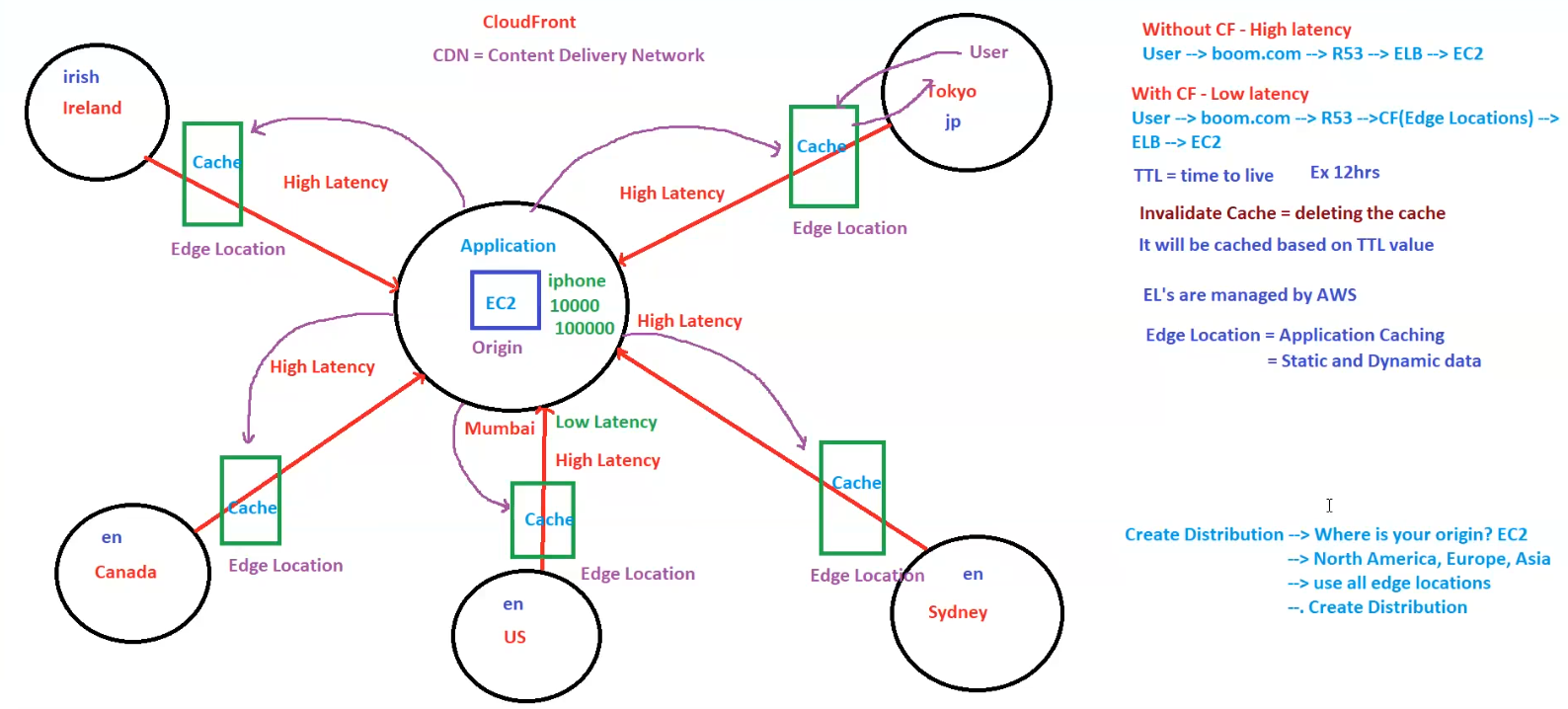
* **Without CloudFront → High Latency**
  + User → boom.com → Route 53 → ELB → EC2
  + Every request goes to the origin server.
* **With CloudFront → Low Latency**
  + User → boom.com → Route 53 → CloudFront (Edge Location) → ELB → EC2
  + Frequently accessed data is served from the nearest **Edge Location (EL)** instead of hitting the origin server.

**4. CloudFront Distribution Setup**

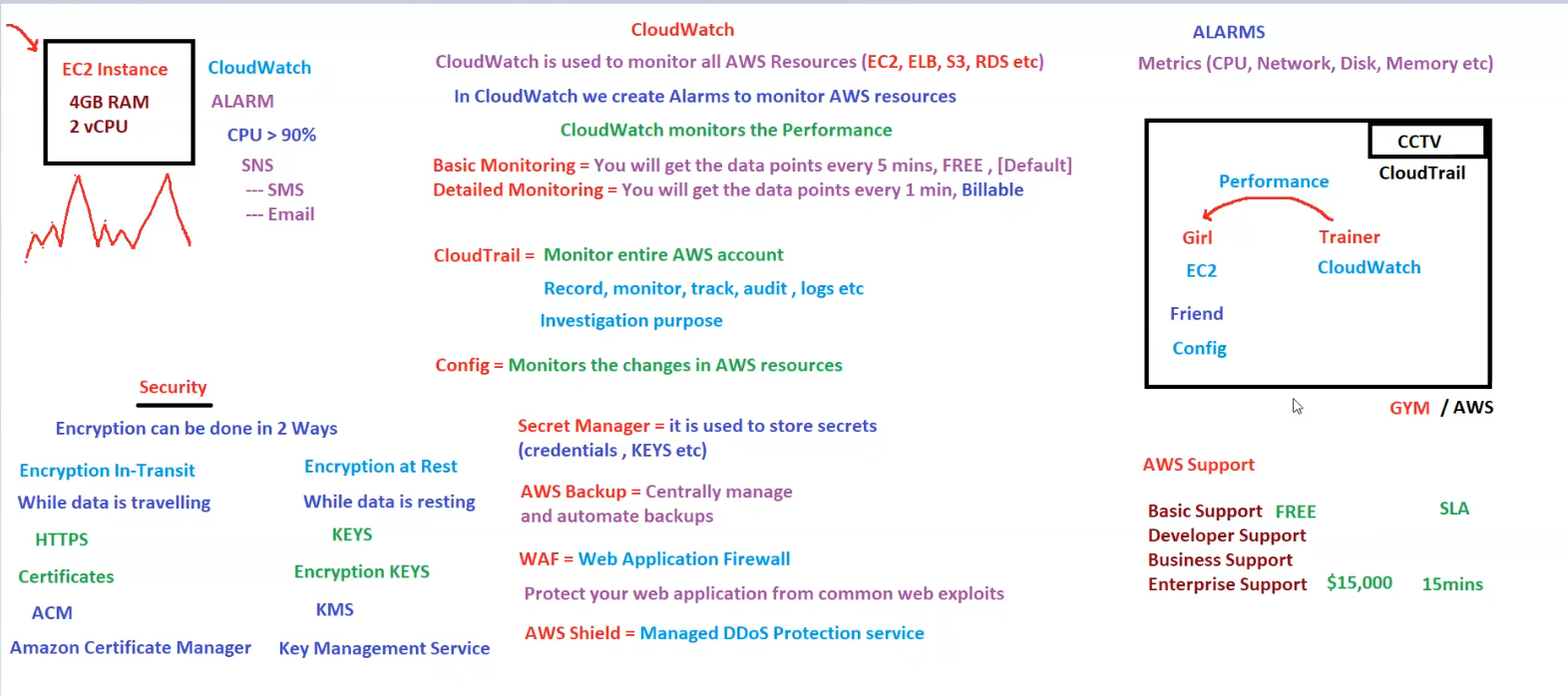
* **Create Distribution**:
  + **Define Origin** (e.g., EC2 instance).
  + **Specify Regions** (e.g., North America, Europe, Asia).
  + **Use all Edge Locations**.
  + **Create CloudFront Distribution**.

**Key Takeaways**

* **Route 53** is AWS's **global** DNS service.
* **CloudFront (CDN)** helps reduce **latency & improves performance**.
* **Edge Locations cache data** for faster delivery.
* **TTL determines caching duration** (default: 12 hours).
* **CloudFront should be used for both static & dynamic content**.



## AWS Monitoring, Security, and Support Overview



**AWS Monitoring, Security, and Support Overview**

**1. AWS CloudWatch (Performance Monitoring)**

* **CloudWatch is used to monitor AWS resources** (e.g., EC2, ELB, S3, RDS).
* **Creates alarms** to monitor resource metrics (CPU, memory, disk, network, etc.).
* **Monitors performance** in two modes:
  + **Basic Monitoring** (Free) – Data every **5 minutes** (default).
  + **Detailed Monitoring** (Billable) – Data every **1 minute**.
* **Alarms can trigger actions** via **SNS (Simple Notification Service)**:
  + Sends notifications via **SMS or Email**.

**2. AWS CloudTrail (Audit & Logging)**

* **Monitors the entire AWS account**.
* **Tracks, records, audits, and logs user activities**.
* Useful for **security investigation & compliance**.

**3. AWS Config (Change Monitoring)**

* **Monitors changes in AWS resources**.
* Helps track **configuration history**.

**4. AWS Security Features**

**Encryption**

* **Two types of encryption**:
  + **Encryption In-Transit** (while data is moving):
    - **Uses HTTPS, SSL certificates**.
    - Managed via **Amazon Certificate Manager (ACM)**.
  + **Encryption At Rest** (for stored data):
    - Uses **Encryption Keys (KMS - Key Management Service)**.

**Other Security Services**

* **Secrets Manager** – Securely stores credentials, API keys, etc.
* **AWS Backup** – Automates backups for AWS resources.
* **WAF (Web Application Firewall)** – Protects web applications from attacks.
* **AWS Shield** – **Managed DDoS protection service**.

**5. AWS Support Plans**

* **Basic Support** – **Free**.
* **Developer Support** – Paid.
* **Business Support** – Paid.
* **Enterprise Support** – **$15,000**, **15-minute SLA**.

**6. Analogy: AWS as a Gym**

* **CloudWatch = Trainer** (Monitors EC2 performance).
* **CloudTrail = CCTV** (Tracks activity logs).
* **Config = Friend** (Notifies about changes in AWS resources).

**Key Takeaways**

* **CloudWatch** is used for **real-time monitoring & alerts**.
* **CloudTrail** is for **logging and auditing user activities**.
* **AWS Config** tracks changes in AWS resources.
* **AWS provides encryption, firewall protection, and backup solutions**.
* **Support plans vary from Free to Enterprise ($15,000/month)**.