# CSC 304 : CLOUD COMPUTING

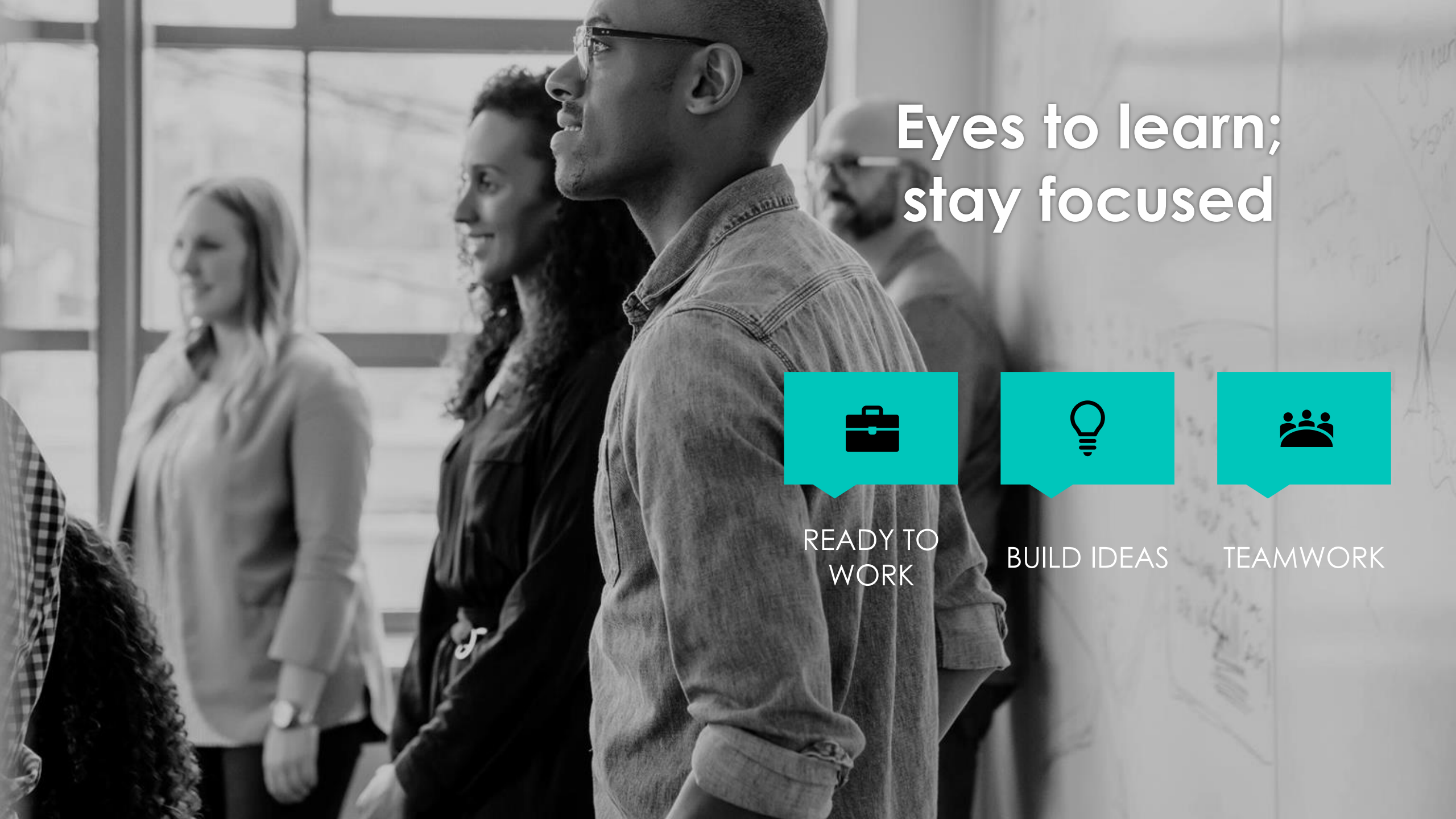
ENOCH O. DANIEL

# COURSE DESCRIPTION

- This course introduces students to the fundamental concepts, principles, and technologies of cloud computing.
- Students will learn about various cloud service models, deployment models, and cloud computing architectures.
- You will also gain hands-on experience with popular cloud platforms and tools.
- Through a combination of lectures, discussions, and practical exercises, students will develop the skills necessary to design, deploy, and manage cloud-based solutions.

# COURSE OBJECTIVES

1. Understand the concepts, principles, and benefits of cloud computing.
2. Identify different cloud service models and deployment models.
3. Explore various cloud computing architectures and design patterns.
4. Gain practical experience with popular cloud platforms and tools.
5. Develop skills for designing, deploying, and managing cloud-based solutions.
6. Analyze security, privacy, and ethical issues related to cloud computing.


Eyes to learn;
stay focused
READY TO WORK
BUILD IDEAS
TEAMWORK

# MODULE 1

Introduction to Cloud Computing

# Evolution and History of Cloud Computing

- **1960s-1970**:
- - The concept of cloud computing traces back to the 1960s and 1970s with the development of time-sharing and mainframe computing.
- - During this period, large mainframe computers were shared among multiple users, allowing them to access computing resources remotely via terminals.

# Evolution and History of Cloud Computing

- **1980s-1990s**
- - The emergence of the internet in the 1980s laid the groundwork for distributed computing.
- - The advent of virtualization technology(**technology use to create virtual representations of servers, storage, networks, and other physical machines**) in the 1990s enabled multiple virtual machines to run on a single physical server, increasing resource utilization and flexibility.
- - The concept of utility computing, where computing resources are provided on-demand like electricity or water, began to gain traction.

# Evolution and History of Cloud Computing

- **2000s**
- - In the early 2000s, companies such as Amazon and Salesforce started offering web-based services, laying the foundation for cloud computing.
- - Amazon Web Services (AWS) was launched in 2006, offering scalable computing resources as services over the internet.
- - Google introduced Google Apps in 2006, providing productivity software delivered over the web.
- - The term "cloud computing" gained popularity as a metaphor for the internet, representing the abstraction of complex infrastructure and services.

# Evolution and History of Cloud Computing

- **2010s:**
- - The 2010s saw widespread adoption of cloud computing across industries, driven by factors such as cost-efficiency, scalability, and flexibility.
- - **Major tech companies including Microsoft (Azure), Google (Google Cloud Platform), and IBM (IBM Cloud) entered the cloud market, offering a range of services and solutions**.
- - The rise of mobile computing and the Internet of Things (IoT) further fueled the demand for cloud services to support data storage, processing, and analytics.
- - Cloud computing became integral to digital transformation initiatives, enabling organizations to innovate and compete in a rapidly evolving marketplace.

# OVERVIEW OF CLOUD COMPUTING

- Cloud computing refers to the delivery of computing services (including servers, storage, databases, networking, software, analytics, and intelligence) over the internet ("the cloud") to offer faster innovation, flexible resources, and economies of scale.
- In essence, cloud computing enables users to access computing resources and services on-demand without the need for owning and maintaining physical hardware or infrastructure.

# DEFINITIONS OF CLOUD COMPUTING

- **A Gartner report :** "... A style of computing in which scalable and elastic IT-enabled capabilities are delivered as a service to external customers using Internet technologies."
- This is a slight revision of Gartner's original definition from 2008, in which
- "Massively scalable" was used instead of "scalable and elastic." This acknowledges the importance of scalability in relation to the ability to scale vertically and not just to enormous proportions.
- **Forrester Research as:** "...a standardized IT capability (services, software, or infrastructure) delivered via Internet technologies in a pay-per-use, self-service way."

# ❑ DEFINITIONS OF CLOUD COMPUTING

❑ **THE NIST DEFINITION OF CLOUD COMPUTING**

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.

# ❑ DEFINITIONS OF CLOUD COMPUTING

❑ **SUMMARY DEFINITION OF CLOUD COMPUTING**

***"Cloud computing is a specialized form of distributed computing that introduces utilization models for remotely provisioning scalable and measured resources.***

# BUSINESS DRIVERS FOR CLOUD COMPUTING DEMANDS

# INTRODUCTION

- The origins and inspirations of many of the characteristics, models, and mechanisms covered throughout this course can be traced back to the upcoming business drivers.

- It is important to note that these influences shaped clouds and the overall cloud computing market from both ends. They have motivated organizations to adopt cloud computing in support of their business automation requirements.

- They have correspondingly motivated other organizations to become providers of cloud environments and cloud technology vendors in order to create and meet the demand to fulfill consumer needs.

# 1. CAPACITY PLANNING

- Capacity planning is the process of determining and fulfilling future demands of an organization's IT resources, products, and services. Within this context, *capacity* represents the maximum amount of work that an IT resource is capable of delivering in a given period of time.

- A discrepancy between the capacity of an IT resource and its demand can result in a system becoming either inefficient (over-provisioning) or unable to fulfill user needs (under-provisioning). Capacity planning is focused on minimizing this discrepancy to achieve predictable efficiency and performance.

# CAPACITY PLANNING

Different capacity planning strategies exist:

- *Lead Strategy* – adding capacity to an IT resource in anticipation of demand
- *Lag Strategy* – adding capacity when the IT resource reaches its full capacity
- *Match Strategy* – adding IT resource capacity in small increments, as demand increases

# CAPACITY PLANNING

- Planning for capacity can be challenging because it requires estimating usage load fluctuations.
- There is a constant need to balance peak usage requirements without unnecessary over-expenditure on infrastructure.
- An example is outfitting IT infrastructure to accommodate maximum usage loads which can impose unreasonable financial investments. In such cases, moderating investments can result in under-provisioning, leading to transaction losses and other usage limitations from lowered usage thresholds.

# 2. COST REDUCTION

A direct alignment between IT costs and business performance can be difficult to maintain. The growth of IT environments often corresponds to the assessment of their maximum usage requirements.

This can make the support of new and expanded business automations an ever-increasing investment. Much of this required investment is funneled into infrastructure expansion because the usage potential of a given automation solution will always be limited by the processing power of its underlying infrastructure.

Two costs need to be accounted for:

- **the cost of acquiring new infrastructure, and**
- **the cost of its ongoing ownership.**

- Operational overhead represents a considerable share of IT budgets, often exceeding up-front investment costs.

# 2. COST REDUCTION

Common forms of infrastructure-related operating overhead include the following:

1. • **technical personnel required to keep the environment operational**
2. **upgrades and patches that introduce additional testing and deployment cycles**
3. • **utility bills and capital expense investments for power and cooling**
4. • **security and access control measures that need to be maintained and enforced to protect infrastructure resources**
5. • **administrative and accounts staff that may be required to keep track of licenses and support arrangements**

# 2. COST REDUCTION

- The on-going ownership of internal technology infrastructure can encompass burdensome responsibilities that impose compound impacts on corporate budgets.
- An IT department can consequently become a significant—and at times overwhelming—drain on the business, potentially inhibiting its responsiveness, profitability, and overall evolution.

# 3. ORGANIZATIONAL AGILITY

- Businesses need the ability to adapt and evolve to successfully face change caused by both internal and external factors. **Organizational agility is the measure of an organization's responsiveness to change**.

- **An IT enterprise often needs to respond to business change by scaling its IT resources beyond the scope of what was previously predicted or planned for.**

- For example, infrastructure may be subject to limitations that prevent the organization from responding to usage fluctuations even when anticipated **if previous capacity planning efforts were restricted by inadequate budgets.**

# 3. ORGANIZATIONAL AGILITY

In other cases**, changing business needs and priorities may require IT resources to be more available and reliable than before**.

Even if sufficient infrastructure is in place for an organization to **support anticipated usage volumes, the nature of the usage may generate runtime exceptions that bring down hosting servers.**

Due to a **lack of reliability controls within the infrastructure**, responsiveness to consumer or customer requirements may be reduced to a point whereby a business' overall continuity is threatened.

# 3. ORGANIZATIONAL AGILITY

**On a broader scale;**

- the up-front investments and infrastructure ownership costs that are required to enable new or expanded business automation solutions may themselves be prohibitive enough for a business to settle for IT infrastructure of less-than-ideal quality, thereby decreasing its ability to meet real-world requirements.

- Worse yet, the business may decide against proceeding with an automation solution altogether upon review of its infrastructure budget, because it simply cannot afford to. This form of inability to respond can inhibit an organization from keeping up with market demands, competitive pressures, and its own strategic business goals.

BREAK;
General discussion

- CLUSTERING
- GRID COMPUTING
- VIRTUALIZATION

# BASIC TERMINOLOGIES IN CLOUD COMPUTING

# CLOUD

**A cloud refers to a distinct IT environment that is designed for the purpose of remotely provisioning scalable and measured IT resources.**

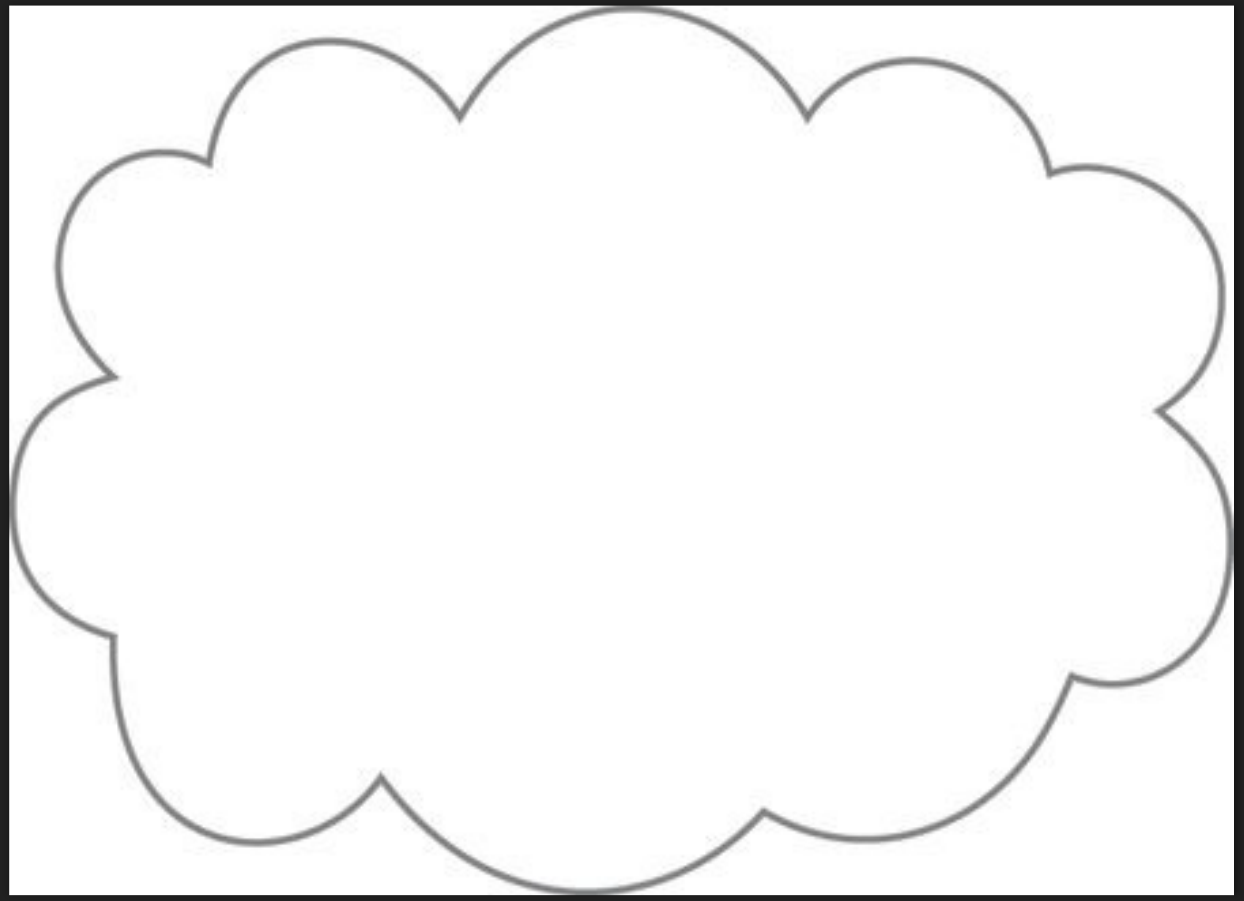
The term originated as a metaphor for the Internet which is, in essence, a network of networks providing remote access to a set of decentralized IT resources.

Prior to cloud computing becoming its own formalized IT industry segment, the symbol of a cloud was commonly used to represent the Internet in a variety of specifications and mainstream documentation of Web-based architectures.

# CLOUD SYMBOL

## CLOUD VS INTERNET

- It is important to distinguish the term “cloud” and the cloud symbol from the Internet.
- There are many individual clouds that are accessible via the Internet.
- Whereas the Internet provides open access to many Web-based IT resources, a cloud is typically privately owned and offers access to IT resources that is metered.



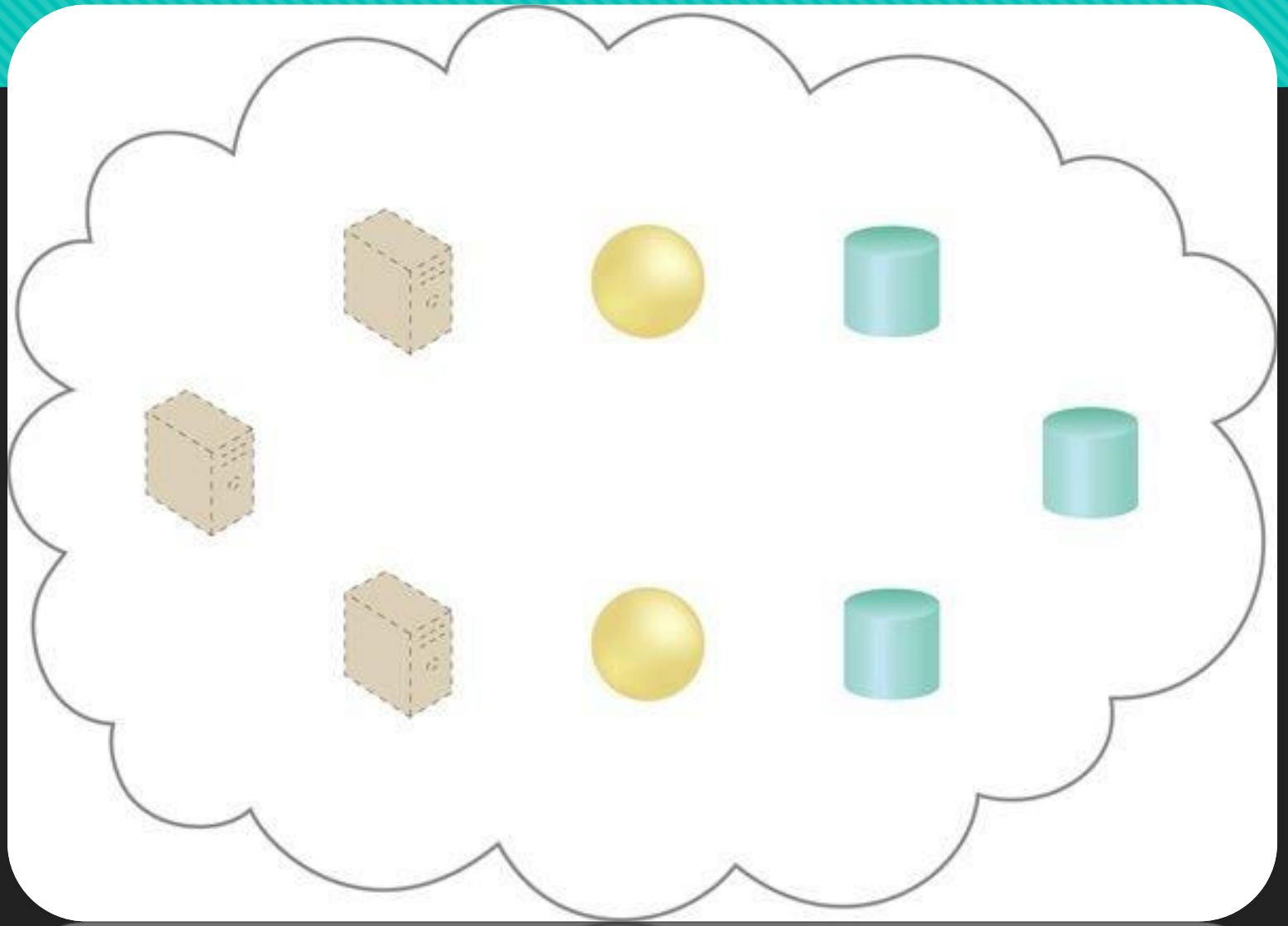
# IT RESOURCES

An IT resource is a physical or virtual IT-related artifact that can be either software-based, such as a virtual server or a custom software program, or hardware-based, such as a physical server or a network device

# IT RESOURCES

# IT RESOURCES

- This diagram illustrates how the cloud symbol can be used to define a boundary for a cloud-based environment that hosts and provisions a set of IT resources.
- The displayed IT resources are consequently considered to be cloud-based IT resources.



# ON-PREMISE

- **An IT resource that is hosted in a conventional IT enterprise within an organizational boundary (that does not specifically represent a cloud) is considered to be located on the premises of the IT enterprise, or on-premise for short**.
- In other words, the term "on-premise" is another way of stating "on the premises of a controlled IT environment that is not cloud-based." This term is used to qualify an IT resource as an alternative to "cloud-based." An IT resource that is on-premise cannot be cloud-based, and vice-versa.

# ON-PREMISE

Note the following key points:

- · **An on-premise IT resource can access and interact with a cloud-based IT resource.**
- • **An on-premise IT resource can be moved to a cloud, thereby changing it to a cloud-based IT resource.**
- • **Redundant deployments of an IT resource can exist in both on-premise and cloud-based environments**.

# Reliability

- Reliability refers to the ability of a system or component to perform its intended functions consistently and accurately under specific conditions for a specified period.
- A reliable system ensures that it delivers consistent performance without unexpected failures or errors, meeting user expectations and requirements.
- Reliability is often measured using metrics such as Mean Time Between Failures (MTBF), which calculates the average time elapsed between system failures, and Mean Time to Repair (MTTR), which measures the average time required to repair a failed system.

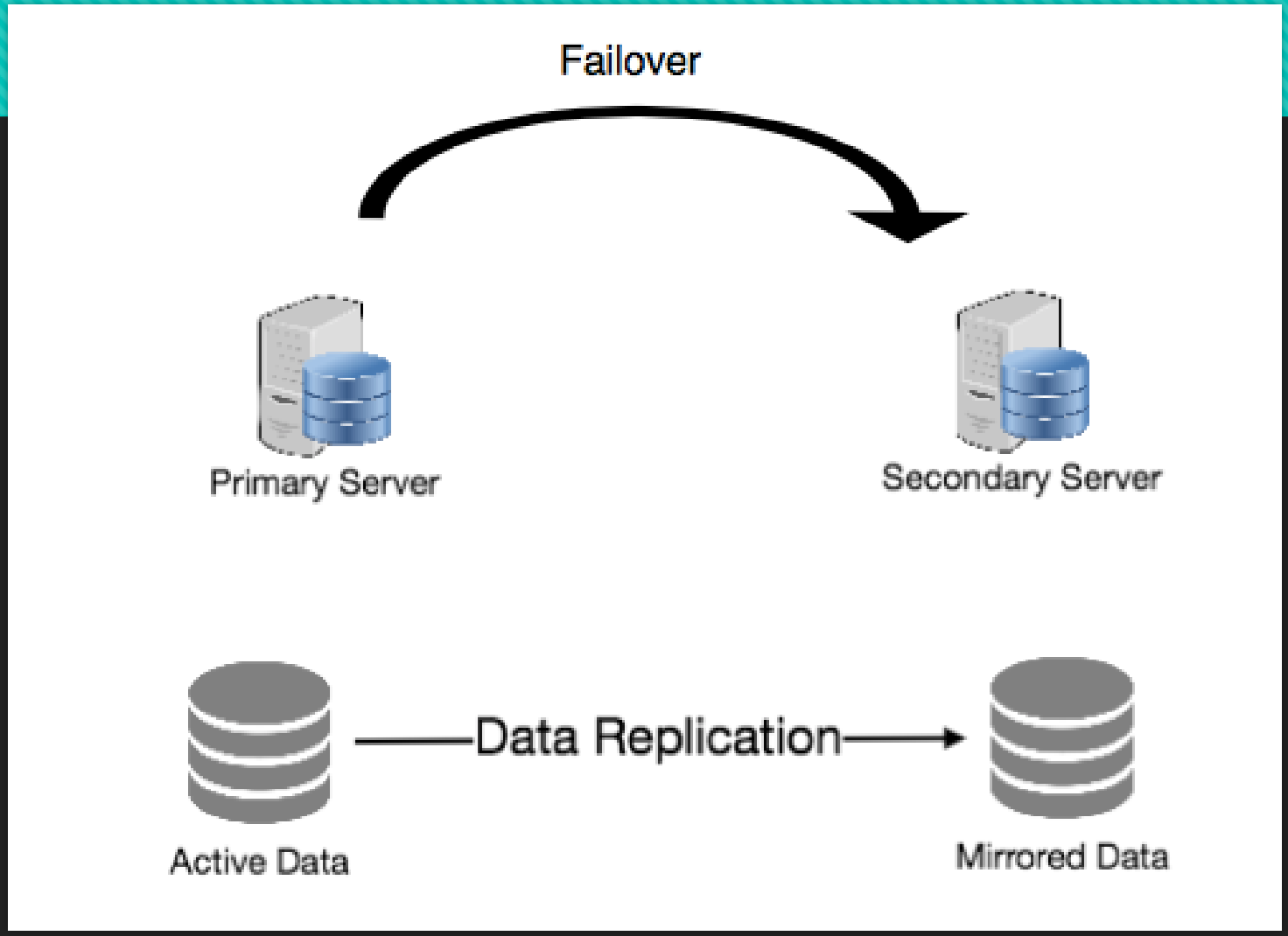
# Availability

- Availability refers to the proportion of time that a system or service is operational and accessible for use.
- Availability ensures that users can access and utilize a system or service when needed, minimizing downtime and disruptions to operations.
- Availability is often expressed as a percentage of uptime over a specific period, typically measured annually. For example, a system with 99.9% availability means it is operational and accessible 99.9% of the time, equating to approximately 8.76 hours of downtime per year.

# Redundancy

- Redundancy refers to the duplication of critical components or systems within a system architecture to provide backup functionality and ensure continuity of operations.
- Redundancy enhances reliability, availability, and fault tolerance by mitigating the impact of component failures **and reducing single points of failure within a system.**
- Redundancy can be implemented at various levels, including hardware redundancy (e.g., redundant power supplies, disk arrays), software redundancy (e.g., hot standby servers, failover clusters), and network redundancy (e.g., redundant network paths, load balancers).

# Redundancy


Failover
Primary Server
Secondary Server
Data Replication
Active Data
Mirrored Data

# Fault Tolerance

- Fault tolerance is the ability of a system to continue operating and providing services in the event of a failure or malfunction of one or more IT components.
- Fault tolerance ensures that a system remains operational and resilient to failures, maintaining service continuity and preventing data loss or service disruptions.
- Fault tolerance is achieved through redundancy, replication, and error detection mechanisms. Redundant components or systems are used to provide backup functionality, allowing the system to switch to alternative resources seamlessly in case of failure.

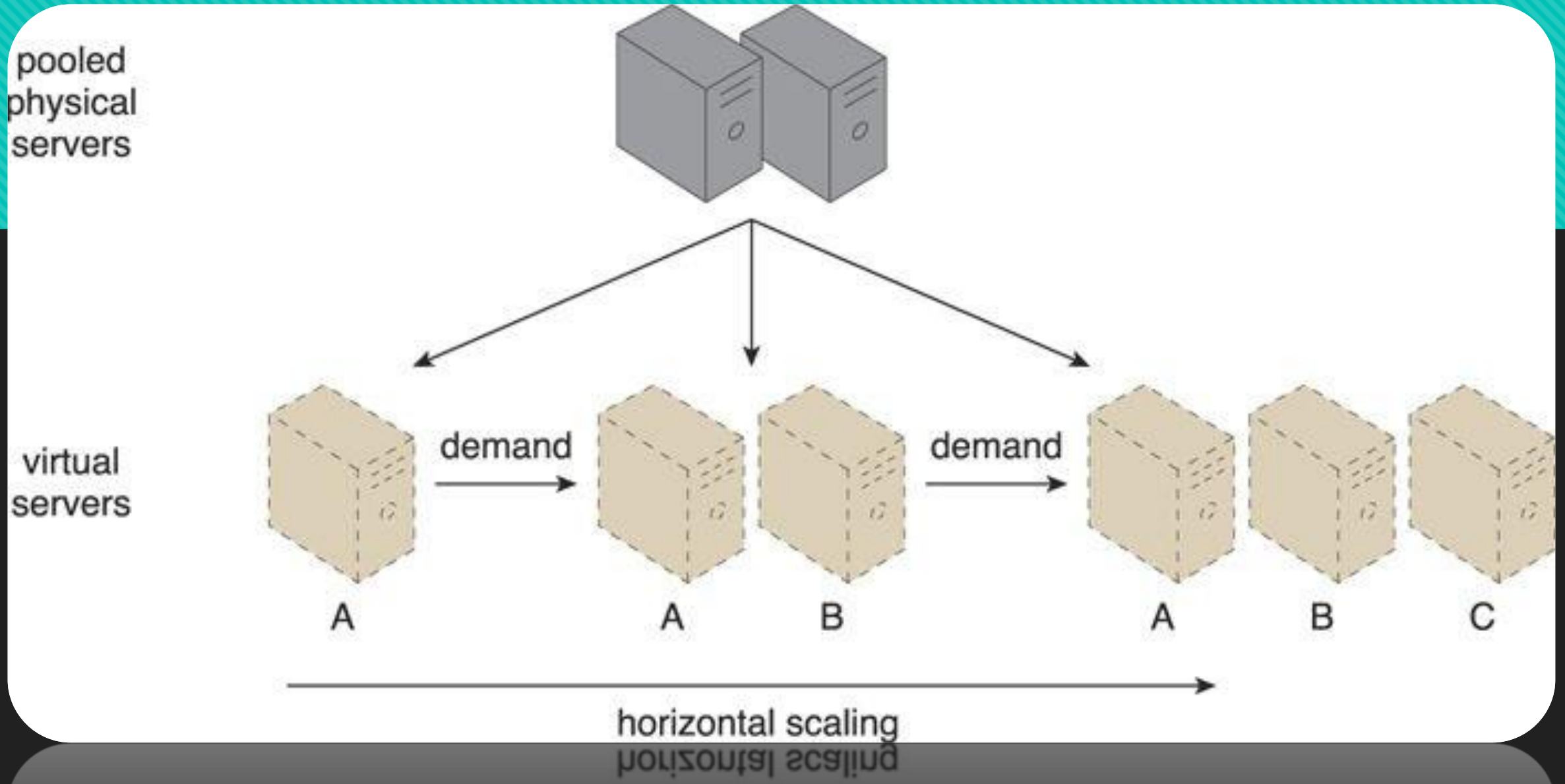
# SCALING

- Scaling, from an IT resource perspective, represents the ability of the IT resource to handle increased or decreased usage demands.

- The following are types of scaling:
  - **Horizontal Scaling – scaling out and scaling in**
  - **Vertical Scaling – scaling up and scaling down**

# HORIZONTAL SCALING

- The allocating or releasing of IT resources that are of the same type is referred to as horizontal scaling.
- The horizontal allocation of resources is referred to as scaling out
- the horizontal releasing of resources is referred to as scaling in.
- Horizontal scaling is a common form of scaling within cloud environments.




**An IT resource (Virtual Server A) is scaled out by adding more of the same IT resources (Virtual Servers B and C).**

# HORIZONTAL SCALING

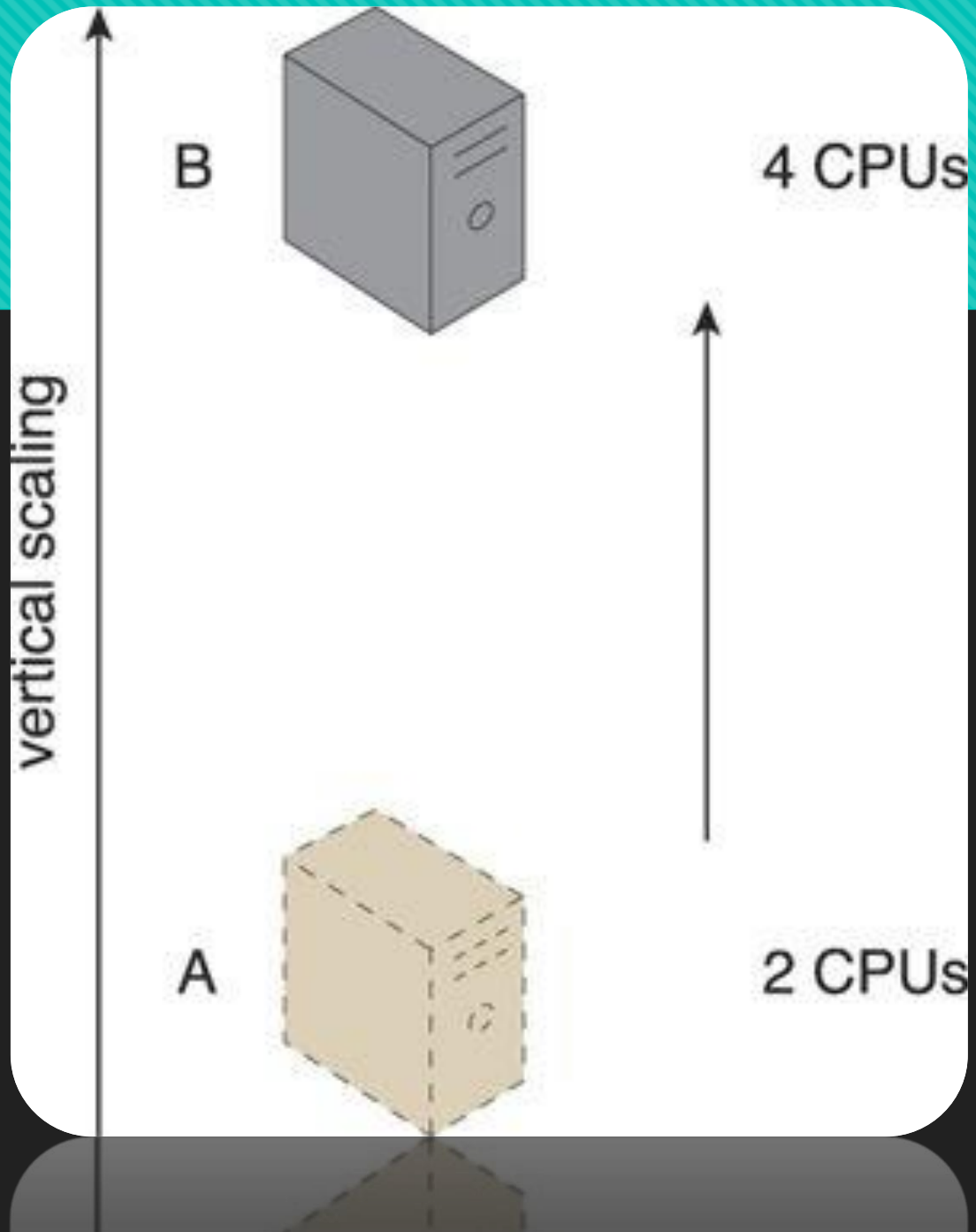
- Horizontal scaling enhances system **reliability and availability by distributing workload across multiple instances.**
- Failures or outages affecting individual instances have a reduced impact on overall system performance, as other instances can continue to handle incoming requests.

# HORIZONTAL SCALING USE CASE

- Horizontal scaling is well-suited for modern cloud-native applications, web servers, microservices architectures, and distributed databases.
- It is particularly beneficial for workloads with unpredictable or fluctuating traffic patterns, as it provides flexibility and agility in resource allocation.

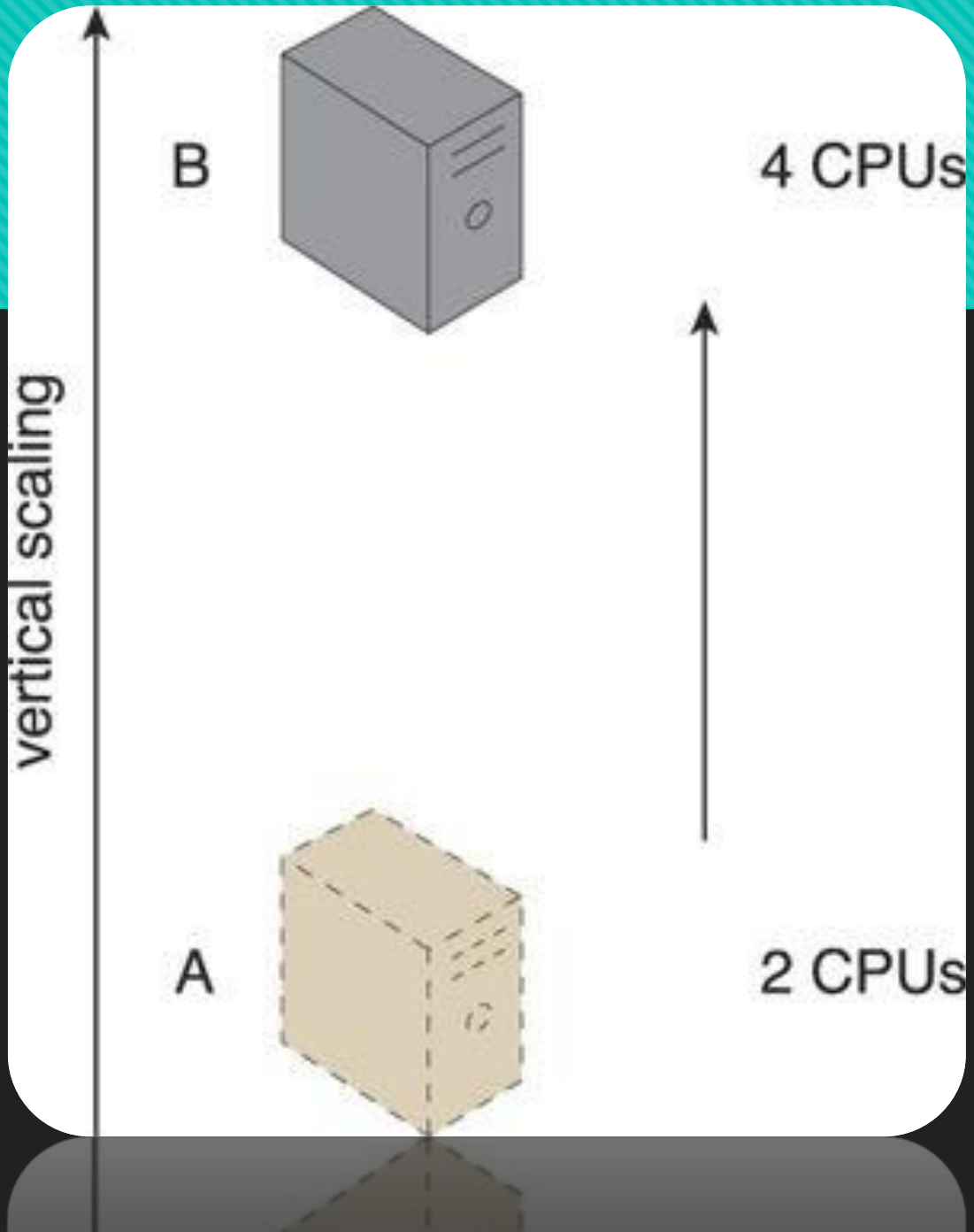
# VERTICAL SCALING

- Vertical scaling, also known as scaling up or scaling vertically, involves increasing the capacity or power of individual hardware resources within a single server or virtual machine.
- This typically involves adding more CPUs, memory (RAM), storage, or other resources to the existing server to handle increased workload demands.



# VERTICAL SCALING

- An IT resource (a virtual server with two CPUs) is scaled up by replacing it with a more powerful IT resource with increased capacity for data storage (a physical server with four CPUs).



# VERTICAL SCALING

- When an existing IT resource is replaced by another with higher or lower capacity, vertical scaling is considered to have occurred.
- Specifically, the replacing of an IT resource with another that has a higher capacity is referred to as scaling up and the replacing an IT resource with another that has a lower capacity is considered scaling down.
- Vertical scaling is less common in cloud environments due to the downtime required while the replacement is taking place.

# VERTICAL SCALING

- Vertical scaling has inherent limitations in terms of scalability. There is a maximum threshold beyond which further vertical scaling becomes impractical or cost-prohibitive.
- Additionally, vertical scaling does not inherently improve **fault tolerance or high availability**, as a single point of failure still exists.

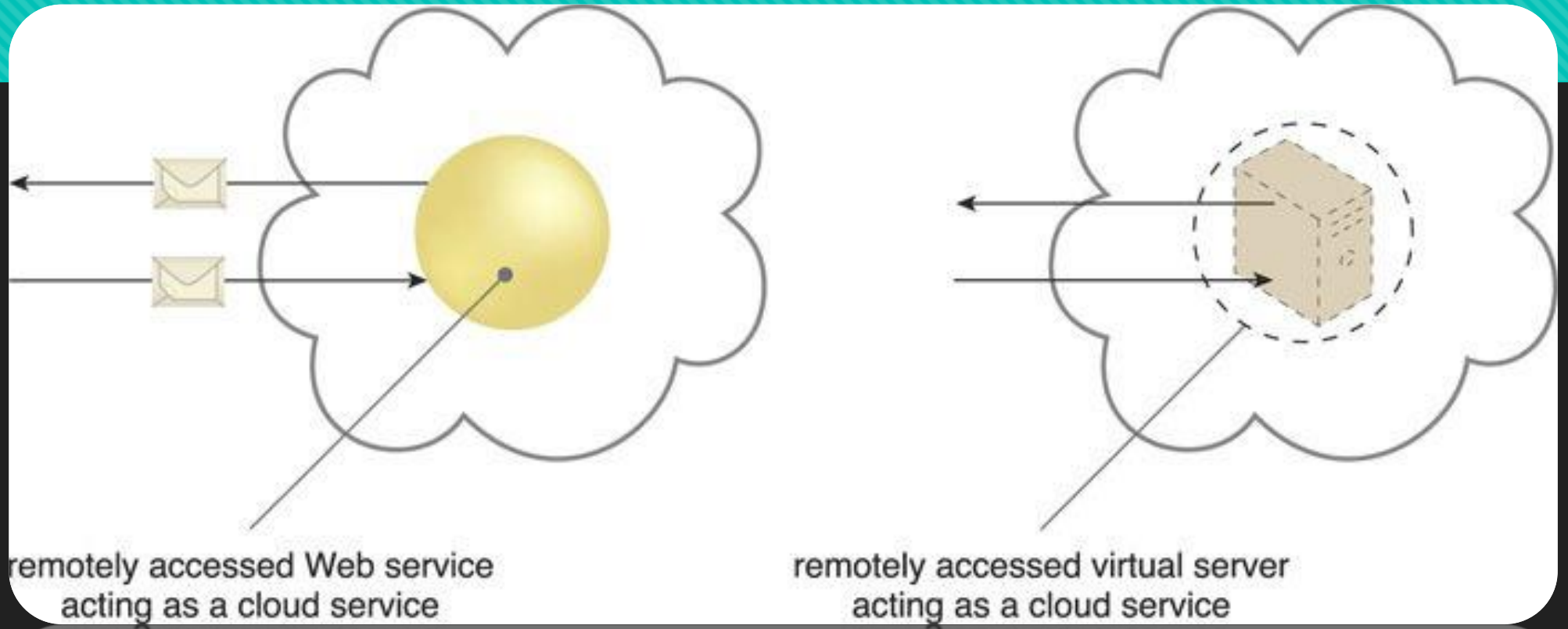
# VERTICAL SCALING USE CASE

- Vertical scaling is commonly used for applications with increasing resource demands over time or for workloads that require high-performance computing capabilities.
- It is often favored for databases, enterprise applications, and legacy systems that are not easily distributed across multiple instances

# CLOUD SERVICE

- A *cloud service* is any IT resource that is made remotely accessible via a cloud. Unlike other IT fields that fall under the service technology umbrella—such as service-oriented architecture—the term “service” within the context of cloud computing is especially broad.
- A cloud service can exist as a simple Web-based software program with a technical interface invoked via the use of a messaging protocol, or as a remote access point for administrative tools or larger environments and other IT resources.

# CLOUD SERVICE



remotely accessed Web service
acting as a cloud service

remotely accessed virtual server
acting as a cloud service

# VERTICAL VS HORIZONTAL SCALING

| Aspect | Horizontal Scaling | Vertical Scaling |
|---|---|---|
| **Resource Allocation** | Resources are distributed across multiple instances, each handling a portion of the workload. | Resources are consolidated within a single server or virtual machine, with increased capacity allocated to handle the entire workload. |
| **Scalability** | Offers virtually unlimited scalability. Additional instances can be added to the system to accommodate growing workloads. | Has inherent limitations in scalability. Further scaling becomes impractical or cost-prohibitive beyond a certain threshold. |
| **Cost** | Can be more cost-effective for handling large-scale workloads, leveraging commodity hardware and allowing incremental resource additions. | May be more costly, as it often involves upgrading hardware components or migrating to higher-tier service plans. |

# VERTICAL VS HORIZONTAL SCALING

| Aspect | Horizontal Scaling | Vertical Scaling |
|---|---|---|
| **Complexity** | Introduces complexity in managing distributed | Generally simpler to implement and manage, |
| | systems, including load balancing, data | involving changes to a single server or VM |
| | consistency, and inter-instance communication. | configuration. |
| **Fault Tolerance** | Enhances fault tolerance by distributing | May have limited fault tolerance, as a failure |
| | workload across multiple instances. Failures | in the single server or VM can lead to downtime |
| | affecting individual instances have reduced | for the entire system. Redundancy and failover |
| | impact on overall system performance. | mechanisms may be needed to mitigate failures. |


CLOUD COMPUTING
DELIVERY MODELS
SAAS
IAAS
PAAS

# DEFINITION

**A *cloud delivery model* represents a specific, pre-packaged combination of IT resources offered by a cloud provider. Three common cloud delivery models have become widely established and formalized:**

1. **• Infrastructure-as-a-Service (IaaS)**
2. **• Platform-as-a-Service (PaaS)**
3. **Software-as-a-Service (SaaS)**

These three models are interrelated in how the scope of one can encompass that of another

# IAAS

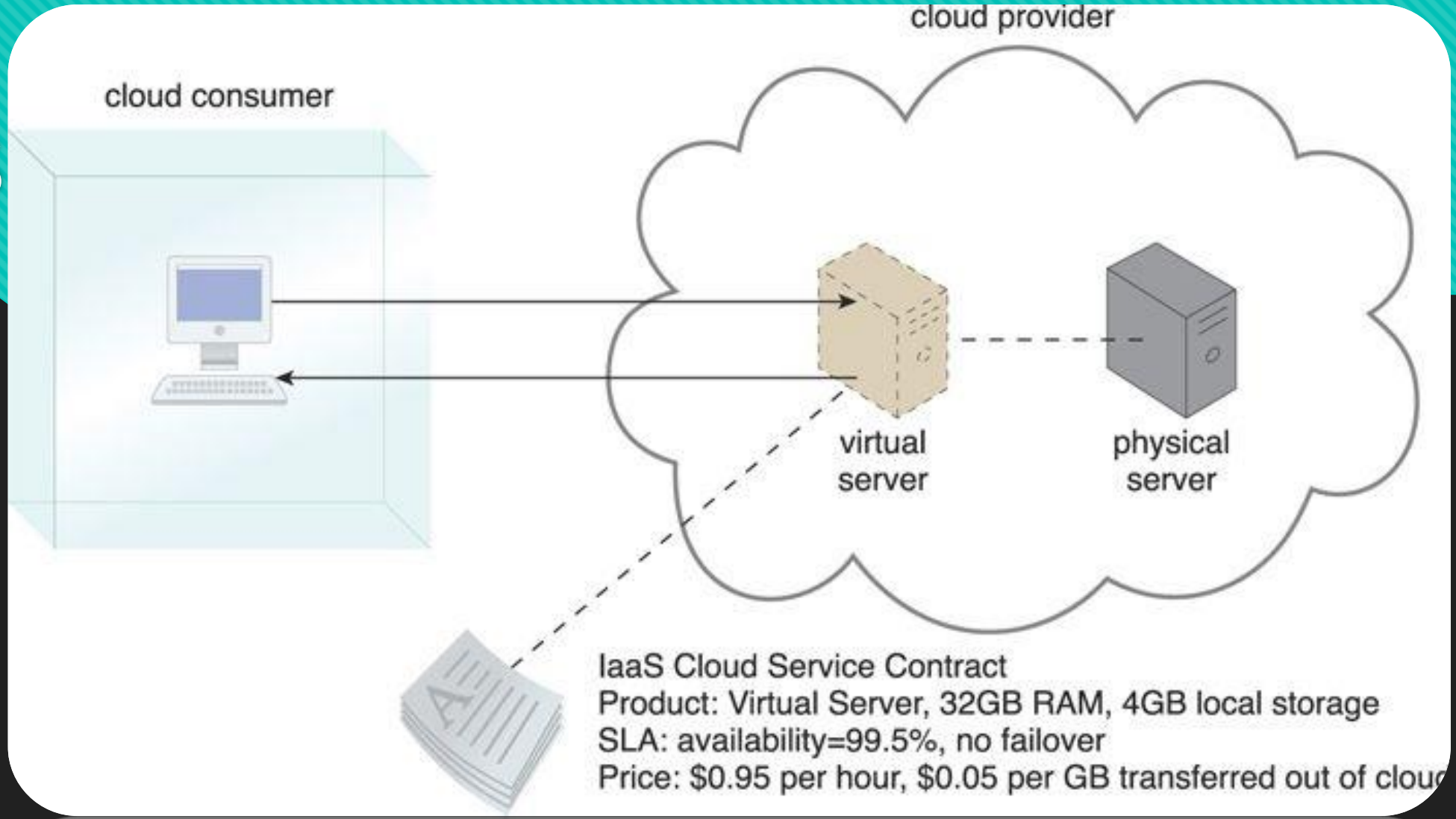
The IaaS delivery model represents a self-contained IT environment comprised of **infrastructure-centric IT resources** that can be accessed and managed via cloud service-based interfaces and tools. This environment can include

1. **hardware,**
2. **network,**
3. **connectivity,**
4. **operating systems,**
5. **and other "raw" IT resources.**

# IAAS

- IT resources are typically virtualized and packaged into bundles that simplify up-front runtime scaling and customization of the infrastructure.
- In general, In this model, **cloud providers offer virtualized computing resources over the internet. Users can rent virtual machines, storage, and networking infrastructure on a pay-as-you-go basis**
- **The general purpose of an IaaS environment is to provide cloud consumers with a high level of control and responsibility over its configuration and utilization**. The IT resources provided by IaaS are generally not pre-configured, placing the administrative responsibility directly upon the cloud consumer..

# IAAS



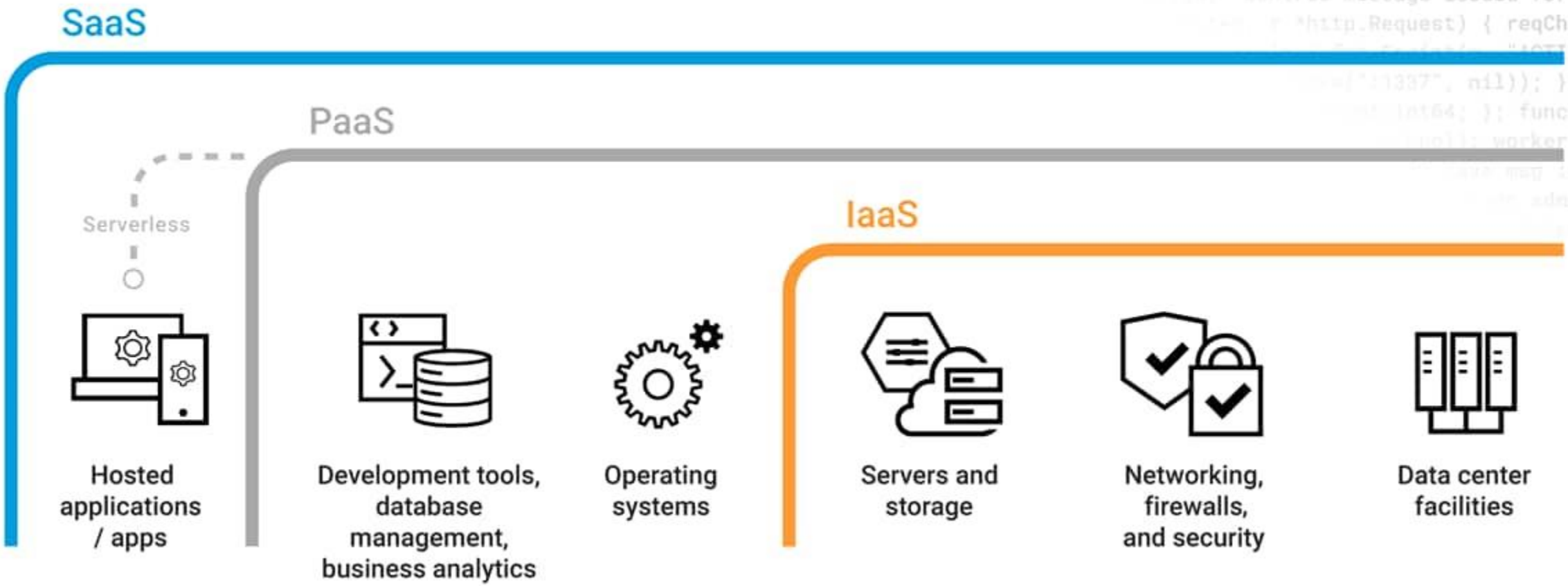

central and primary IT resource within a typical IaaS environment is the virtual server. Virtual servers are leased by specifying server hardware requirements, such as processor capacity, memory, and local storage space, as shown above

# PAAS

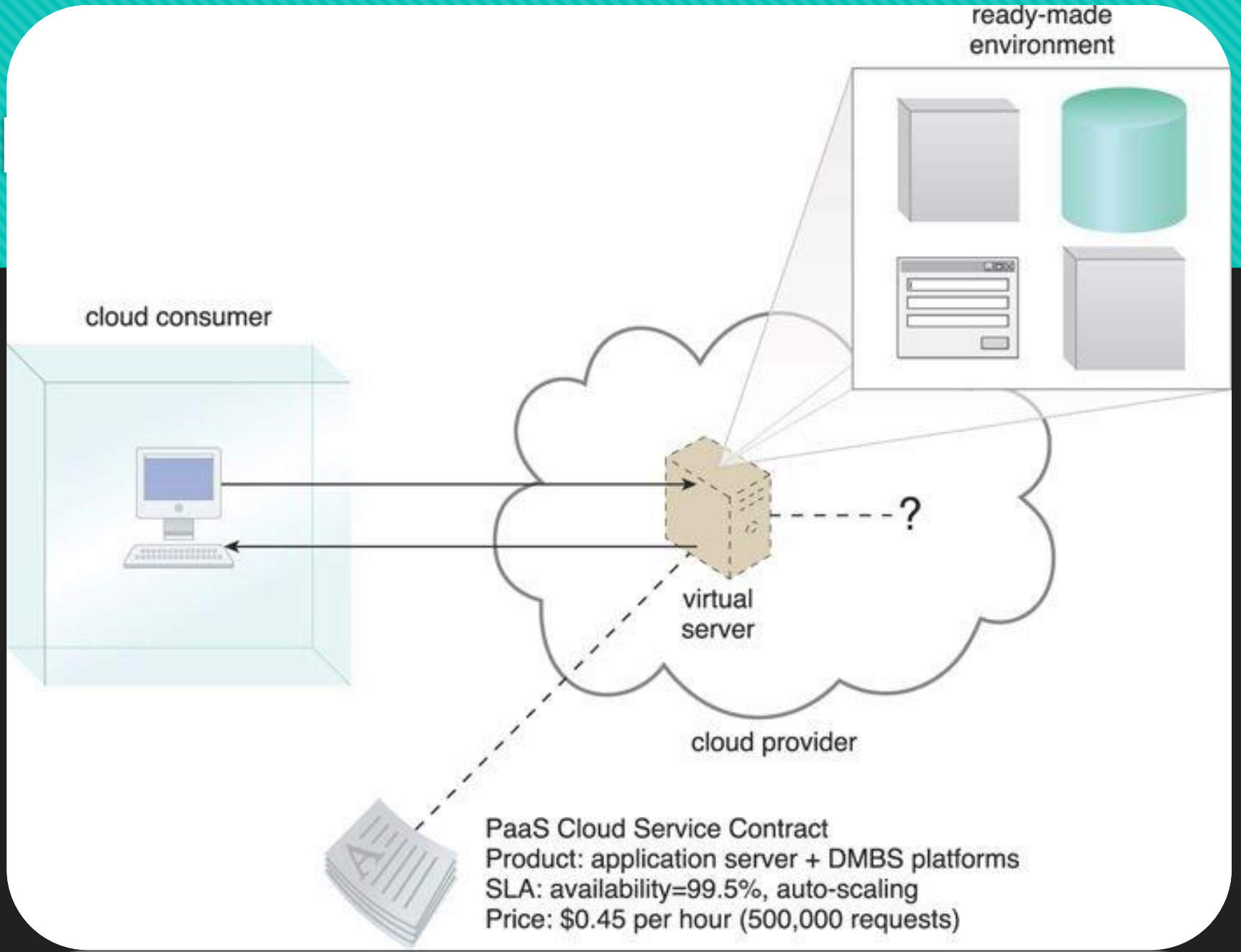
- The PaaS delivery model represents a pre-defined "ready-to-use" environment typically comprised of already deployed and configured IT resources.
- **PaaS provides a platform allowing customers to develop, run, and manage applications without dealing with the underlying infrastructure. It typically includes tools and services for application development, such as programming languages, databases, and development frameworks**.
- Specifically, PaaS relies on (and is primarily defined by) the usage of a ready made environment that establishes a set of pre-packaged products and tools used to support the entire delivery lifecycle of custom applications

# PAAS

- Platform as a Service (PaaS) typically provides a comprehensive platform for developers to build, deploy, and manage applications without worrying about the underlying infrastructure.
- PaaS offerings often include runtime environments, development tools, middleware, databases, and other services required for application development and deployment.


SaaS
PaaS
Serverless
IaaS
Hosted applications / apps
Development tools, database management, business analytics
Operating systems
Servers and storage
Networking, firewalls, and security
Data center facilities

# PAAS ARC


ready-made
environment
cloud consumer
?
virtual
server
cloud provider
PaaS Cloud Service Contract
Product: application server + DMBS platforms
SLA: availability=99.5%, auto-scaling
Price: $0.45 per hour (500,000 requests)

# PAAS PROVIDERS

- 1.Heroku - Heroku is a cloud platform that simplifies application deployment with support for multiple languages, automated scaling, CI/CD pipelines, and a rich ecosystem of add-ons and integrations.

- 2.Google App Engine - Google App Engine is a fully managed platform for building web applications and APIs, offering automatic scaling, built-in security, and seamless integration with other Google Cloud services.

# PAAS PROVIDERS

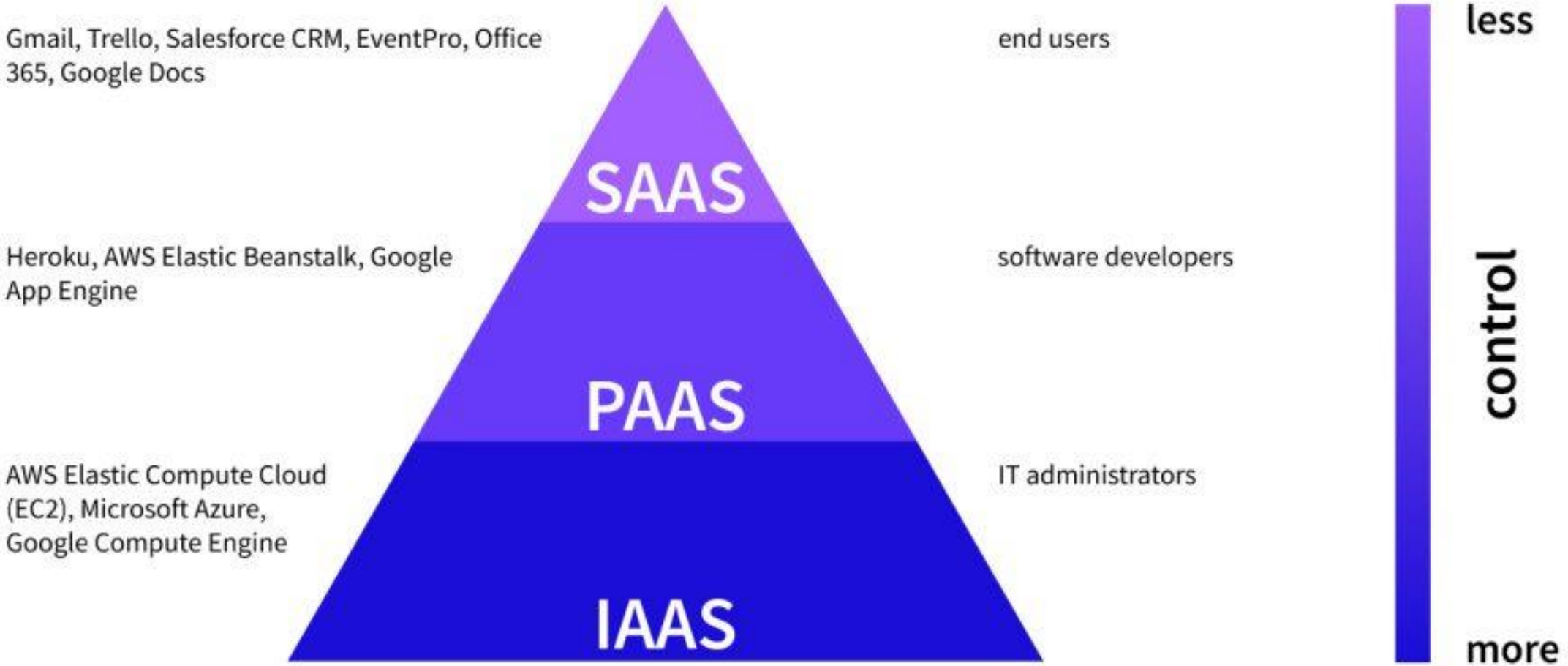
- 3. Microsoft Azure App Service - enables developers to build, deploy, and scale web applications and APIs with support for multiple languages, automatic scaling, CI/CD integration, and built-in monitoring.
- 4. AWS Elastic Beanstalk - AWS Elastic Beanstalk simplifies application deployment on AWS infrastructure with support for multiple languages, automatic provisioning, scaling, and monitoring of underlying resources.

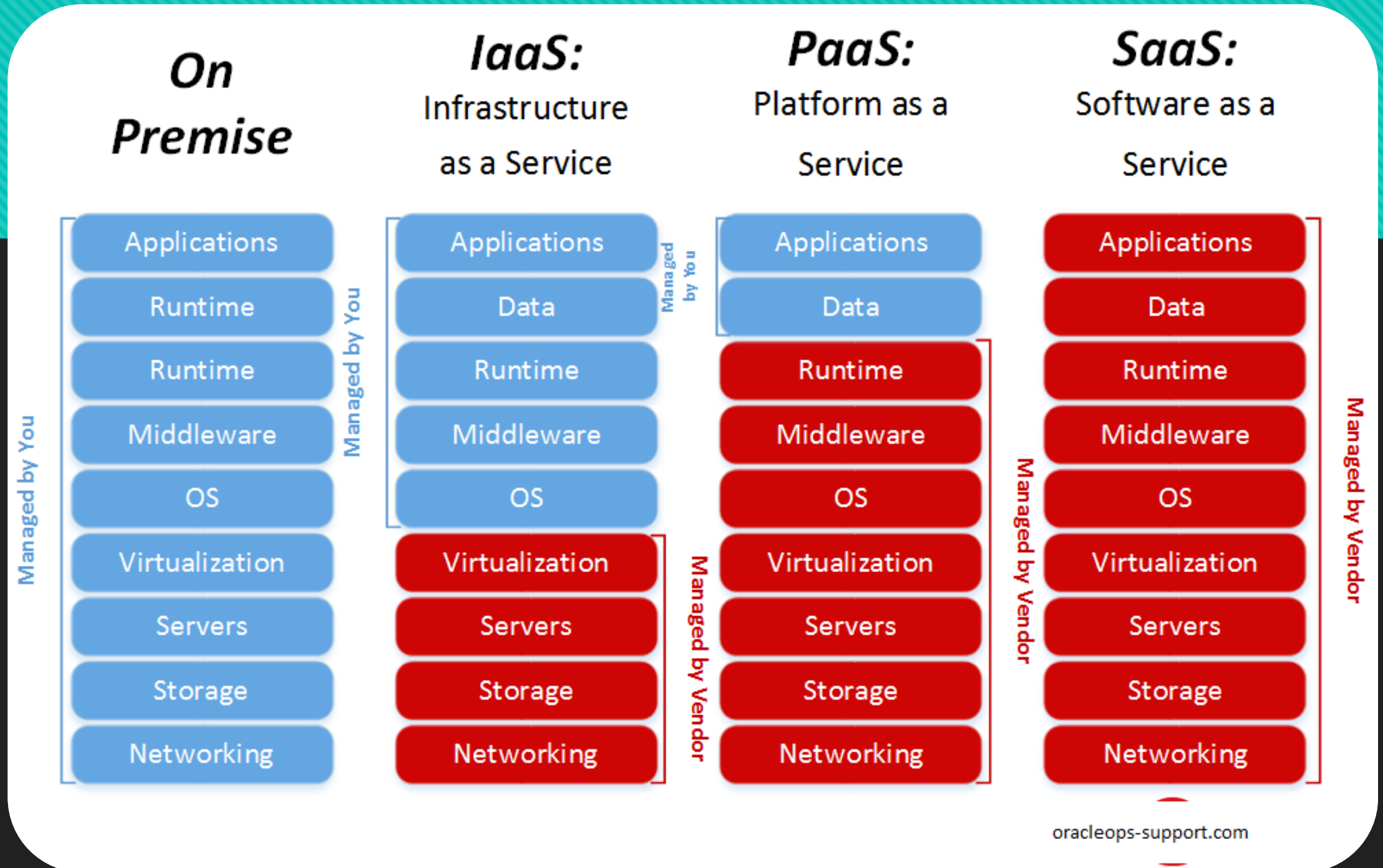
# PAAS VS SERVERLESS COMPUTING

- serverless computing, often referred to as Function as a Service (FaaS), focuses specifically on executing code in response to events or triggers without the need for developers to manage server infrastructure.
- While serverless computing is a form of cloud computing, it's more specific than traditional PaaS offerings.
- With serverless computing, developers typically upload functions or code snippets to the platform, and the platform handles scaling, provisioning, and managing the infrastructure required to execute those functions.

# SAAS

- A software program positioned as a shared cloud service and made available as a “product” or generic utility represents the typical profile of a SaaS offering. The SaaS delivery model is typically used to make a reusable cloud service widely available (often commercially) to a range of cloud consumers.
- This cloud services are  software applications offered over the internet on a subscription basis. Users can access these applications through a web browser without needing to install or maintain any software locally.
- Common examples of SaaS applications include Google Workspace (formerly G Suite), Microsoft Office 365, Salesforce, and Dropbox.


Gmail, Trello, Salesforce CRM, EventPro, Office 365, Google Docs
end users
SAAS
Heroku, AWS Elastic Beanstalk, Google App Engine
software developers
PAAS
AWS Elastic Compute Cloud (EC2), Microsoft Azure, Google Compute Engine
IT administrators
IAAS
less
control
more


On Premise
IaaS: Infrastructure as a Service
PaaS: Platform as a Service
SaaS: Software as a Service
Applications
Runtime
Runtime
Middleware
OS
Virtualization
Servers
Storage
Networking
Data
Managed by You
Managed by Vendor
oracleops-support.com

# IAAS VS PAAS VS SAAS

| Cloud Delivery Model | Typical Level of Control Granted to Cloud Consumer | Typical Functionality Made Available to Cloud Consumer |
|---|---|---|
| SaaS | usage and usage-related configuration | access to front-end user-interface |
| PaaS | limited administrative | moderate level of administrative control over IT resources relevant to cloud consumer's usage of platform |
| IaaS | full administrative | full access to virtualized infra-structure-related IT resources and, possibly, to underlying physical IT resources |

# IAAS VS PAAS VS SAAS

| Cloud Delivery Model | Common Cloud Consumer Activities | Common Cloud Provider Activities |
|---|---|---|
| SaaS | uses and configures cloud service | implements, manages, and maintains cloud service<br>monitors usage by cloud consumers |
| PaaS | develops, tests, deploys, and manages cloud services and cloud-based solutions | pre-configures platform and provisions underlying infrastructure, middleware, and other needed IT resources, as necessary<br>monitors usage by cloud consumers |
| IaaS | sets up and configures bare infrastructure, and installs, manages, and monitors any needed software | provisions and manages the physical processing, storage, networking, and hosting required<br>monitors usage by cloud consumers |

# CHARACTERISTICS OF CLOUD COMPUTING

**1. On-Demand Self-Service**

- Users can provision computing resources as needed automatically, without requiring human interaction with each service provider. This characteristic allows for rapid and efficient deployment of services.

**2. Broad Network Access**

- Cloud services are accessible over the network and can be used through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).

**3. Resource Pooling**

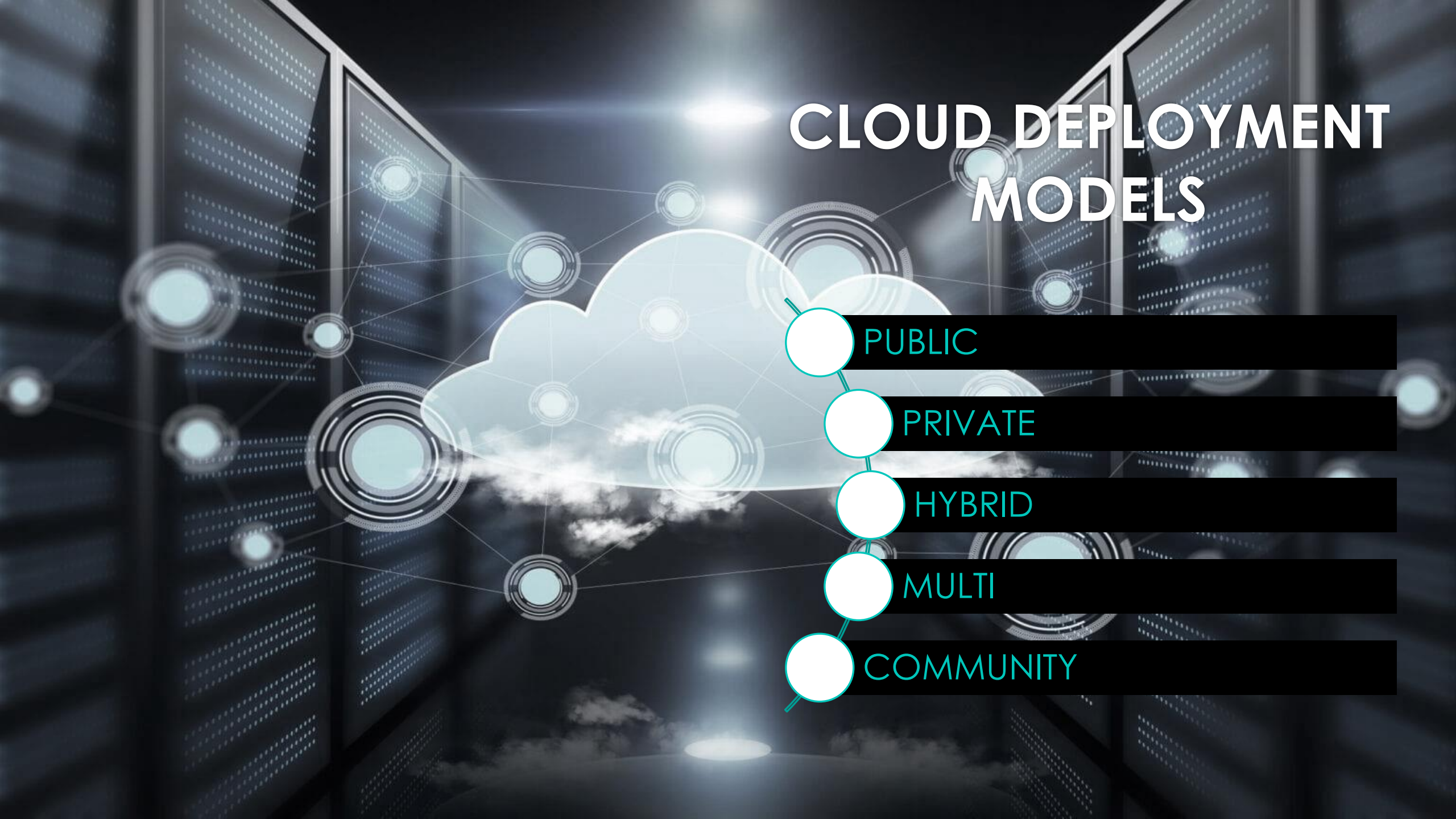
- The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter).

# CHARACTERISTICS OF CLOUD COMPUTING

3. **Resource Pooling**
   - The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter).

4. **Rapid Elasticity**
   - Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.

5. **Measured Service**
   - Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service. This supports a pay-as-you-go model.


CLOUD DEPLOYMENT MODELS
PUBLIC
PRIVATE
HYBRID
MULTI
COMMUNITY

# cloud deployment model

**A cloud deployment model represents a specific type of cloud environment, primarily distinguished by ownership, size, and access.**

There are four common cloud deployment models;

1. • Public cloud
2. • Community cloud
3. • Private cloud
4. • Hybrid

# 1. Public cloud

- A public cloud is a publicly accessible cloud environment owned by a third-party cloud provider. The IT resources on public clouds are usually provisioned via the cloud delivery models and are generally offered to cloud consumers at a cost.
- The IT resources are shared among multiple users.
- They offer scalability, cost-effectiveness, and flexibility. The cloud provider is responsible for the creation and on-going maintenance of and its IT resources
- the public cloud and its IT resources. Examples include AWS, Azure, and Google Cloud Platform

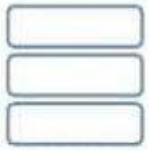
# 2. Private  cloud

- This is a deployment model that is dedicated to a single organization, either managed internally or by a third-party provider. It offers greater control, security, and customization but may require more investment in infrastructure

# 3. Private cloud

- The actual administration of a private cloud environment may be carried out by internal or outsourced staff. With a private cloud, the same organization is technically both the cloud consumer and provider
- In order to differentiate these roles:
- • a separate organizational department typically assumes the responsibility for provisioning the cloud (and therefore assumes the cloud provider role) departments requiring access to the private cloud assume the cloud consumer role

# 2. Pri

| PUBLIC | | PRIVATE |
| --- | --- | --- |
| Publicly shared virtualized resource |  | Privately shared virtualized resource |
| Support multiple customers |  | Cluster of dedicated customers |
| Support Internet connectivity |  | Connectivity over Internet, fibre, & private network |
| Suited for less confidential information |  | Suited for secured confidential information & core systems |

VS

2. P

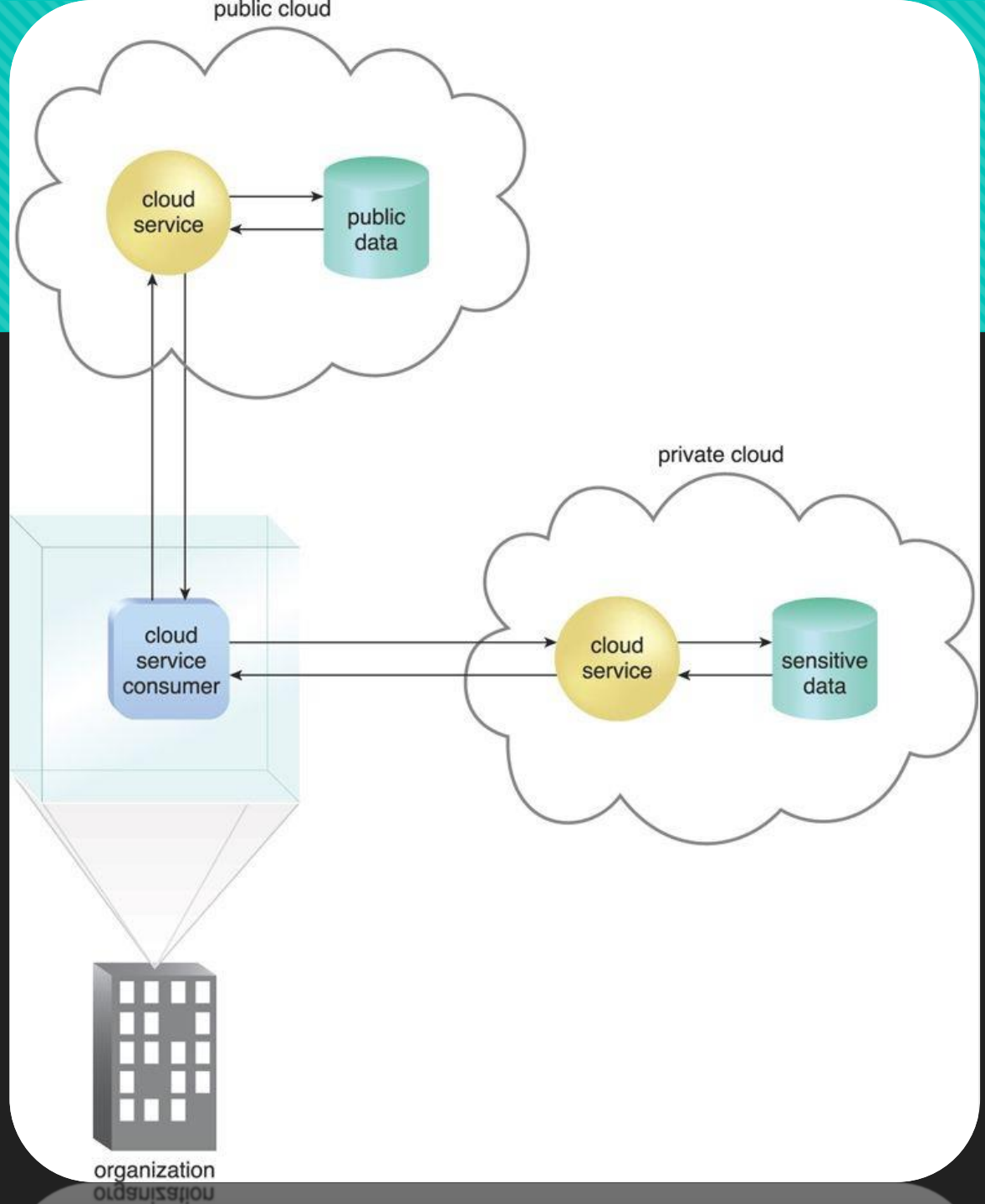
# Private network vs public network

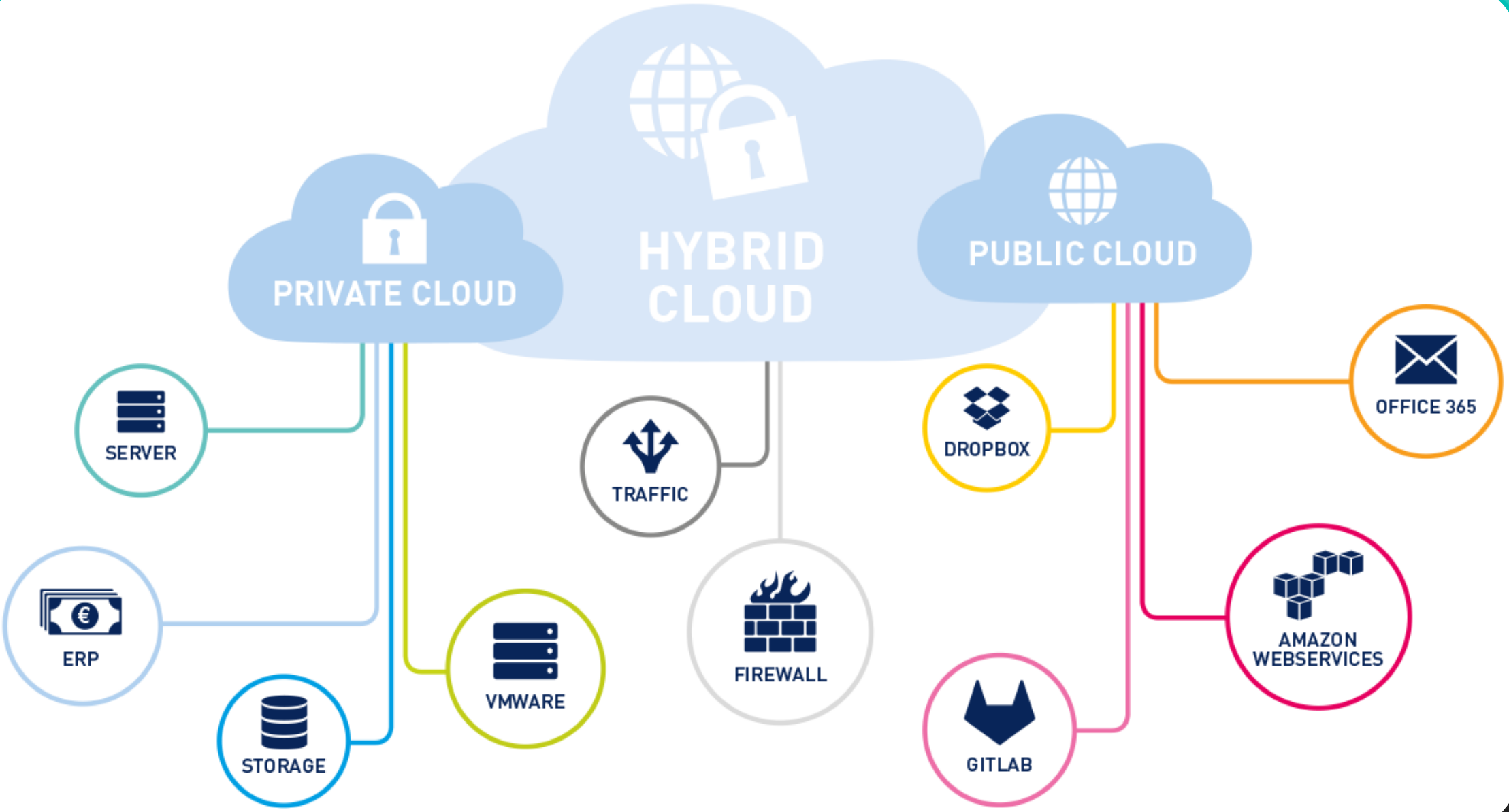
| Difference | Private Network | Public Network |
| --- | --- | --- |
| Accessibility | Restricted to Specific Users | Open and Accessible to Anyone |
| Purpose | Ideal for Internal Communications and Sensitive Data | Used for Broad Interactions and General Connectivity |
| Security | Offers Enhanced Security | Potentially Vulnerable to External Threats |
| IP Addressing | Uses Private IP Address Ranges | Uses Globally Unique IP Addresses |
| Cost | Costlier | Often Cheaper |

# 4. Hybrid cloud

- A hybrid cloud is a cloud environment comprised of two or more different cloud deployment models(public and private ).
- For example, a cloud consumer may choose to deploy cloud services processing sensitive data to a private cloud and other, less sensitive cloud services to a public cloud. The result of this combination is a hybrid deployment model
- It provides flexibility, scalability, and the ability to leverage existing investments while maintaining control over sensitive data.

# 4. Hybrid cloud




HYBRID CLOUD
PRIVATE CLOUD
PUBLIC CLOUD
SERVER
ERP
STORAGE
VMWARE
TRAFFIC
FIREWALL
DROPBOX
GITLAB
OFFICE 365
AMAZON WEBSERVICES

# 4. Multi-Cloud

- Multi-cloud refers to the strategy of using services and resources from multiple cloud providers to meet an organization's computing needs.
- It offers redundancy, risk mitigation, and the ability to leverage specialized services from different providers
- By spreading workloads across multiple cloud platforms, organizations can mitigate the impact of outages, service disruptions, or data breaches associated with a single provider.

# 5. Community cloud

- A community cloud is a deployment model that is limited to a specific community of cloud consumers.
- The community cloud may be jointly owned by the community members or by a third-party cloud provider that provisions a public cloud with limited access.
- Community cloud infrastructure is shared among several organizations with similar requirements, such as regulatory compliance or industry standards. It offers collaboration, cost-sharing, and specialized services tailored to specific communities.


Cloud-Enabling Technology
BROADBOAND INTERNET ARC
VIRTUALIZATION
DATA CENTER
MULTI TENANT TECH

# broadband networks and internet architecture

- broadband networks and internet architecture play crucial roles in enabling the delivery of cloud services and facilitating connectivity between users and cloud provider
- Broadband networks are essential for providing high-speed internet access to users accessing cloud services. They enable users to connect to cloud platforms and access resources such as virtual machines, storage, applications, and databases over the internet

# ISPs

- ISPs, or Internet Service Providers, are companies that provide access to the internet which play a crucial role in providing connectivity between users and cloud service providers. ISPs offer various types of internet connections, including broadband, DSL, cable, fibre-optic etc
- ISPs facilitate the interconnection between their networks and the networks of CSPs through peering and transit agreements. These agreements enable the exchange of internet traffic between users and cloud services, ensuring efficient data transmission and access to cloud resources..
- They  maintain the network infrastructure necessary for delivering internet services, including routers, switches, cables, and data centers. They also manage the routing of internet traffic between different networks.

# Router and switches

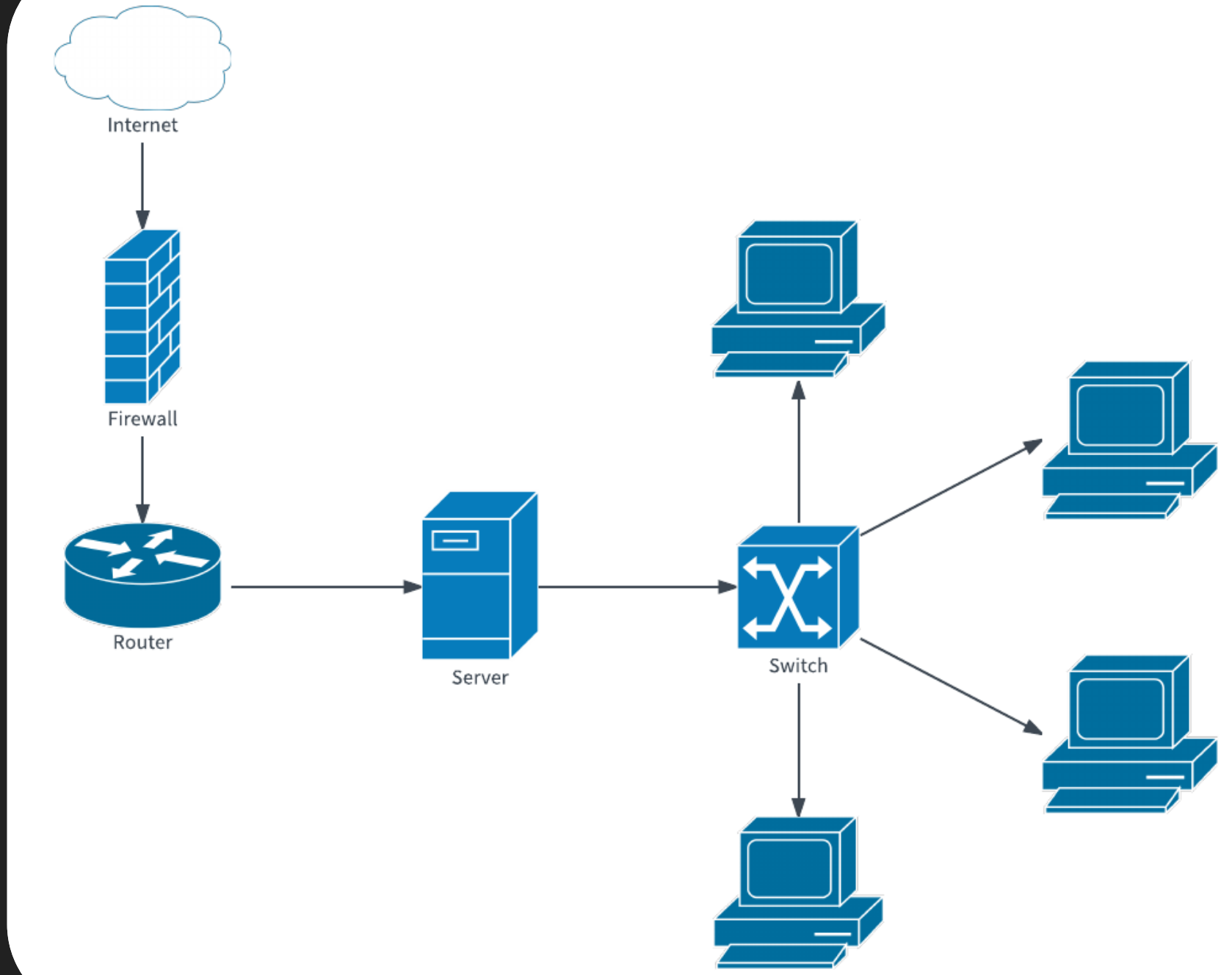
Routers and switches are both networking devices used by ISPs to connect devices within a network, but they serve different purposes :

- **ROUTER**
  1. **A router is a networking device that connects multiple networks and forwards data packets between them.**
  2. It operates at the network layer (Layer 3) of the OSI model and makes decisions based on IP addresses.
  3. Routers use routing tables to determine the best path for forwarding packets to their destination.
  4. They often provide additional features such as firewall protection, NAT (Network Address Translation), and DHCP (Dynamic Host Configuration Protocol) services.

# Routers and switches

- **SWITCH**

1. **A switch is a networking device that connects multiple devices within a single network and forwards data frames between them**.
2. It operates at the data link layer (Layer 2) of the OSI model and makes decisions based on MAC addresses.
3. Switches use MAC(media access control) address tables to determine the appropriate port to forward frames to their destination device. address embedded into the network interface card (NIC) by the manufacturer and is assigned at the factory.
4. They are used to create local area networks (LANs) and improve network performance by reducing network congestion and collisions.
5. Switches are commonly used in homes, businesses, and data centers to connect computers, printers, servers, and other network devices within a LAN.


Internet
Firewall
Router
Server
Switch

# Routers and switches in the cloud

1. **Virtual Private Clouds (VPCs) -**

- - Cloud providers offer Virtual Private Clouds (VPCs), which are logically isolated sections of the cloud where users can deploy their resources.
- - Within a VPC, routers are used to manage traffic between different subnets and route data packets between virtual machines, containers, and other resources.
- - Switches are utilized within VPCs to connect multiple resources within the same subnet and facilitate communication between them.

# Routers and switches in the cloud

3. **Load Balancing-**

- - Load balancers, often implemented as specialized routers or switches, distribute incoming network traffic across multiple servers or instances to optimize resource utilization and ensure high availability and performance.
- - Cloud-based load balancers are integral components of scalable and resilient cloud architectures, efficiently distributing traffic to backend resources.

# Routers and switches in the cloud

4. **Network Security-**

- - Routers and switches play critical roles in network security within cloud environments, implementing access control lists (ACLs), firewall rules, and other security policies to control traffic flow and protect resources from unauthorized access or malicious activity.
- - Virtual routers and switches, often implemented as software-defined networking (SDN) components, provide flexible and scalable security enforcement mechanisms within cloud environments.

# Routers and switches in the cloud

5. **Interconnectivity and Hybrid Clouds -**

- - Routers and switches facilitate connectivity between different cloud regions, availability zones, or cloud providers, enabling the creation of hybrid cloud architectures.
- - They establish secure and reliable connections between on-premises infrastructure and cloud resources, facilitating seamless data transfer and workload migration between environments.

# Routers and switches in the cloud

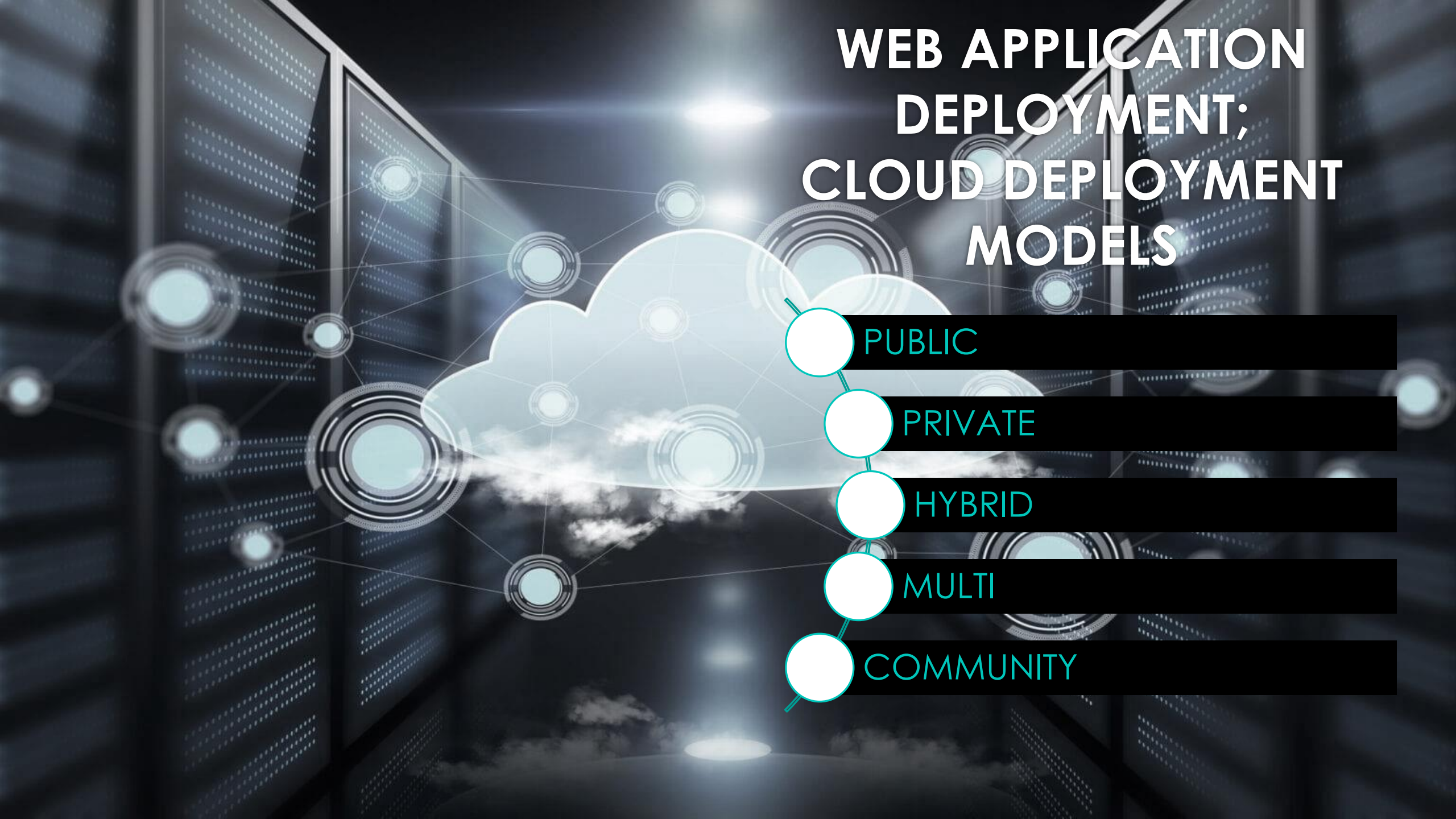
- **SWITCH**

1. **A switch is a networking device that connects multiple devices within a single network and forwards data frames between them**.
2. It operates at the data link layer (Layer 2) of the OSI model and makes decisions based on MAC addresses.
3. Switches use MAC(media access control) address tables to determine the appropriate port to forward frames to their destination device. address embedded into the network interface card (NIC) by the manufacturer and is assigned at the factory.
4. They are used to create local area networks (LANs) and improve network performance by reducing network congestion and collisions.
5. Switches are commonly used in homes, businesses, and data centers to connect computers, printers, servers, and other network devices within a LAN.

# Routers and switches in the cloud

- **SWITCH**

1. **A switch is a networking device that connects multiple devices within a single network and forwards data frames between them**.
2. It operates at the data link layer (Layer 2) of the OSI model and makes decisions based on MAC addresses.
3. Switches use MAC(media access control) address tables to determine the appropriate port to forward frames to their destination device. address embedded into the network interface card (NIC) by the manufacturer and is assigned at the factory.
4. They are used to create local area networks (LANs) and improve network performance by reducing network congestion and collisions.
5. Switches are commonly used in homes, businesses, and data centers to connect computers, printers, servers, and other network devices within a LAN.



# WEB APPLICATION DEPLOYMENT; CLOUD DEPLOYMENT MODELS

- PUBLIC
- PRIVATE
- HYBRID
- MULTI
- COMMUNITY

# AMAZON WEB SERVICES -AWS


- AWS EC2
- AMAZON EBS
- VPC
- AWS S3 BUCKET
- CLOUD DIAGRAMMING

# About AWS

- AWS stands for Amazon Web Services, It is an expanded cloud computing platform provided by Amazon Company. AWS provides a wide range of services with a pay-as-per-use pricing model over the Internet such as Storage, Computing power, Databases, Machine Learning services, and much more.
- AWS facilitates for both businesses and individual users with effectively hosting the applications, storing the data securely, and making use of a wide variety of tools and services improving management flexibility for IT resources

# How AWS Works?

- AWS has its own network infrastructure on establishing the datacenters in different regions mostly all over the world. Its global Infrastructure acts as a backbone for operations and services provided by AWS.
- It facilitates the users on creating secure environments using Amazon VPCs ( Virtual Private Clouds ).
- Essential services like Amazon EC2 and Amazon S3 for utilizing the compute and storage service with elastic scaling. It supports the dynamic scaling of the applications with the services such as Auto Scaling and Elastic Load Balancing ( AWS ELB ).

# Advantages Of Amazon Web Services

- AWS allows you to easily scale your resources up or down as your needs change, helping you to save money and ensure that your application always has the resources it needs.
- AWS provides a highly reliable and secure infrastructure, with multiple data centers and a commitment to 99.99% availability for many of its services.
- AWS offers a wide range of services and tools that can be easily combined to build and deploy a variety of applications, making it highly flexible.
- AWS offers a pay-as-you-go pricing model, allowing you to only pay for the resources you actually use and avoid upfront costs and long-term commitments.


# AWS FUNDAMENTALS – AWS

CLOUD COMPUTING PROVIDER

# AWS CLOUD GLOBAL INFRASTRUCTURE; REGION AND AZs

- The AWS global infrastructure is massive and is divided into geographical regions. The geographical regions are then divided into separate availability zones. While selecting the geographical regions for AWS, three factors come into play

1. Optimizing Latency
2. Reducing cost
3. Government regulations (Some services are not available for some regions).

# AWS CLOUD GLOBAL INFRASTRUCTURE; REGION AND AZs

- **REGION -** An AWS Region is a physical location in the world where we have multiple Availability Zones. AWS provide the services with respective division of regions. The regions are divided based on geographical areas/locations and will establish data centers. Based on need and traffic of users, the scale of data centers is depended to facilitate users with low-latencies of services
- Each region is divided into at least three availability zones that are physically isolated from each other, which provides business continuity for the infrastructure as in a distributed system. **The largest region North Virginia (US-East), has six availability zones. These availability zones are connected by high-speed fiber-optic networking.**

# AWS CLOUD INFRASTRUCTURE, REGION AND AZs

- **AZs** - Availability Zones consist of one or more discrete data centers, each with redundant power, networking, and connectivity, housed in separate facilities.
- These Availability Zones offers the ability to operate production applications and databases that are more highly available, fault tolerant, and scalable than would be possible from a single data center.
- All AZs in an AWS Region are interconnected with high-bandwidth, low-latency networking, over fully redundant, dedicated metro fiber providing high-throughput, low-latency networking between AZs
- Each Availability Zones (AZs) are physically separated locations within a Region that have their own power, cooling, and networking, to further increase fault tolerance and low latency.

## AWS Global Infrastructure Map

105 Availability Zones within 33 geographic regions, with announced plans for 18 more Availability Zones and six more AWS Regions in Malaysia, Mexico, New Zealand, the Kingdom of Saudi Arabia, Thailand, and the AWS European Sovereign Cloud.

**33 launched Regions**
each with multiple Availability Zones

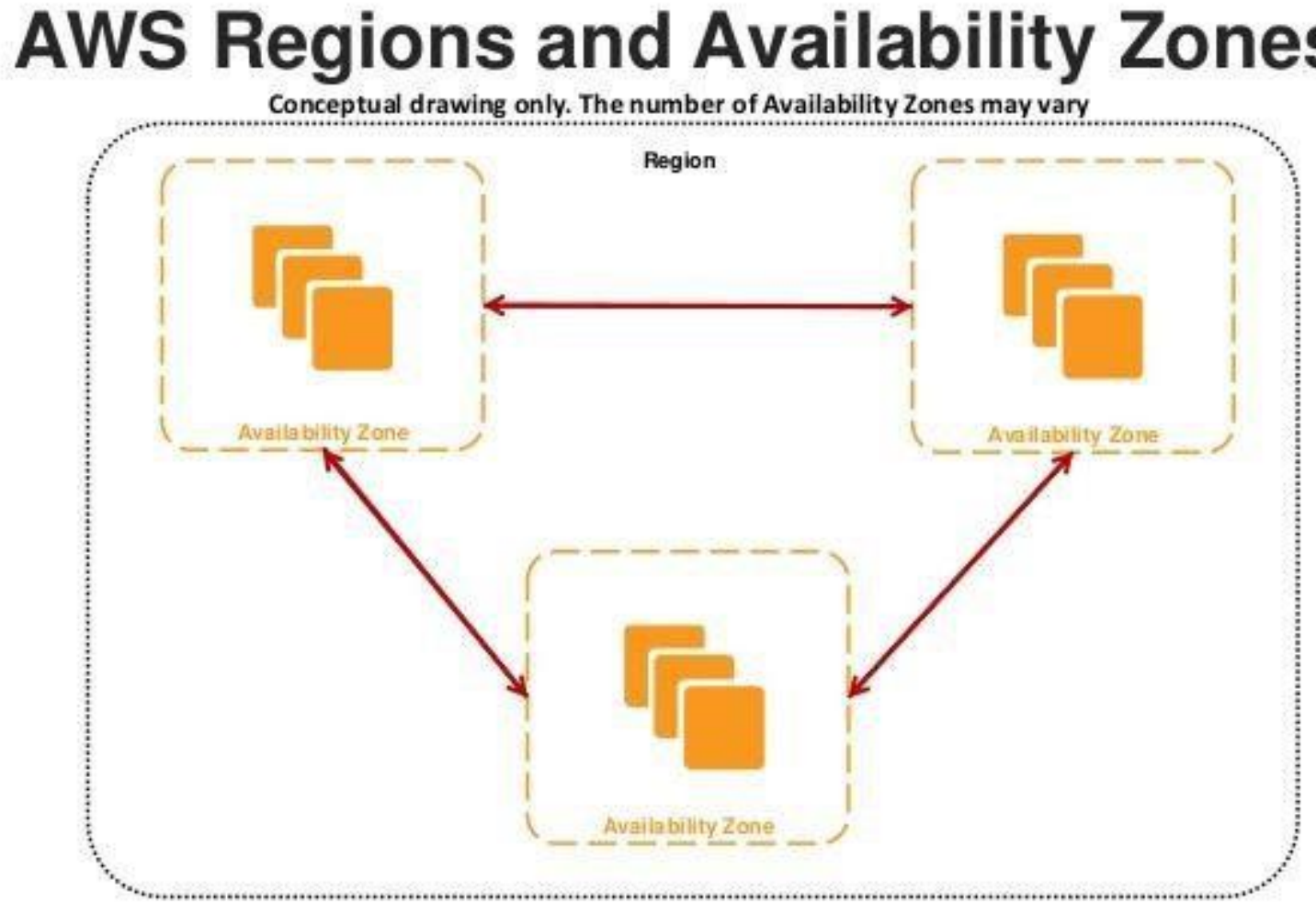
**105 Availability Zones**

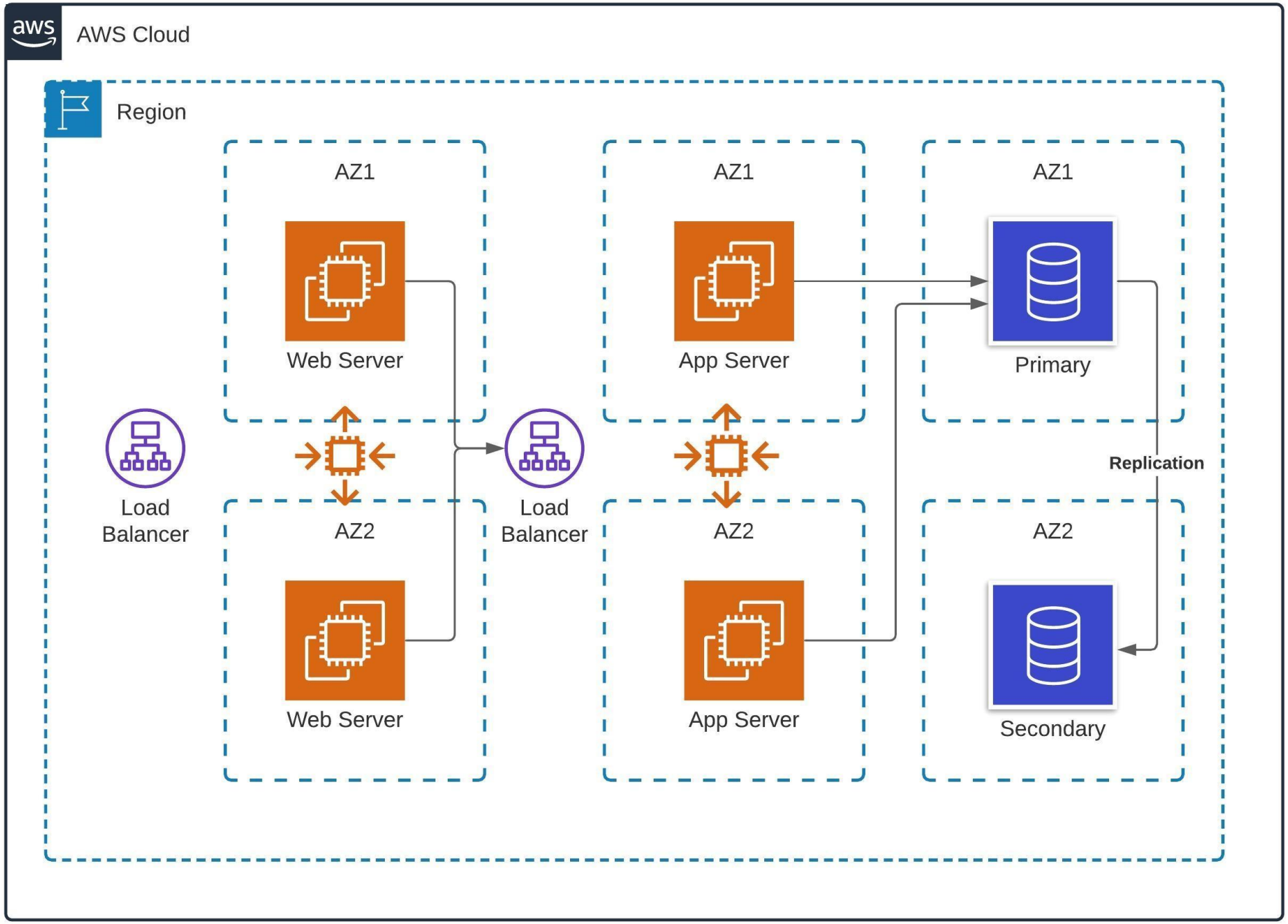
**600+ CloudFront POPs**
and 13 Regional edge caches


## AWS Global Infrastructure Map

105 Availability Zones within 33 geographic regions, with announced plans for 18 more Availability Zones and six more AWS Regions in Malaysia, Mexico, New Zealand, the Kingdom of Saudi Arabia, Thailand, and the AWS European Sovereign Cloud.

# AWS CLOUD INFRASTRUCTURE, REGION AND AZs




AWS Cloud
Region
AZ1
Web Server
AZ1
App Server
AZ1
Primary
Load Balancer
AZ2
Web Server
Load Balancer
AZ2
App Server
Replication
AZ2
Secondary

# AWS REGION & AZs

- Unlike other cloud providers, who often define a region as a single data center, the multiple AZ design of every AWS Region offers advantages for customers.
- Each AZ has independent power, cooling, and physical security and is connected via redundant, ultra-low-latency networks.
- AWS customers focused on high availability can design their applications to run in multiple AZs to achieve even greater fault-tolerance.
- AWS infrastructure Regions meet the highest levels of security, compliance, and data protection.

# AWS CORE SERVICES

- **1. Compute**

Compute services in AWS provide scalable and flexible computing capacity in the cloud, allowing users to run virtual servers and manage computing resources.

- **Example: Amazon EC2 (Elastic Compute Cloud)**

**Amazon EC2** provides resizable compute capacity in the cloud. It allows users to launch and manage virtual machines called instances, configure security and networking, and manage storage. Users can choose from various instance types optimized for different use cases such as compute-optimized, memory-optimized, and storage-optimized instances.

# AWS CORE SERVICES

- **2. Networking**

Networking services in AWS offer a variety of solutions to manage network traffic, establish secure connections, and ensure high availability and performance for applications.

- **Example**: **Amazon VPC (Virtual Private Cloud)**

**Amazon VPC** lets users provision a logically isolated section of the AWS cloud where they can launch AWS resources in a virtual network that they define. Users have complete control over the virtual networking environment, including the selection of IP address ranges, creation of subnets, and configuration of route tables and network gateways.

# AWS CORE SERVICES

**3. Databases**

- **Description**: Database services in AWS provide managed database solutions that support various types of databases, including relational, NoSQL, and data warehousing, ensuring high availability, security, and performance.
- **Example**: **Amazon RDS (Relational Database Service)**

- **Amazon RDS** makes it easy to set up, operate, and scale a relational database in the cloud. It supports several database engines, including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle, and Microsoft SQL Server. RDS handles routine database tasks such as backups, patch management, and hardware provisioning.

# AWS CORE SERVICES

- **4. Storage**

Storage services in AWS offer scalable, durable, and secure storage solutions for a variety of use cases, including backups, data archiving, and application data storage.

- **Example**: **Amazon S3 (Simple Storage Service)**

**Amazon S3** provides scalable object storage with high durability and availability. Users can store and retrieve any amount of data at any time, from anywhere on the web. S3 is designed for 99.999999999% (11 9's) durability and provides comprehensive security and compliance capabilities.

# AWS services

Top AWS Services

- AWS Lambda - It is a service in Serverless Architecture with Function as a Service facilitating serverless computing i.e., running the code on response to the events, the background environment management of servers is handled by aws automatically. It helps the developers to completely focus on the logic of code build.

# ACCESSING AWS SERVICES

The following are several ways to access AWS services, catering to different user preferences, needs, and use cases

- AWS Management Console- This is the web-based graphical user interface (GUI) provided by AWS. It allows users to access and manage AWS services through a browser. Users can navigate through various services, configure settings, and perform tasks using the console.
- AWS Command Line Interface (CLI)- The AWS CLI is a unified tool that provides a command-line interface for managing AWS services. Users can perform tasks such as launching EC2 instances, managing S3 buckets, and configuring security groups directly from the command line. It's particularly useful for scripting and automating tasks.

# ACCESSING AWS services

- AWS Software Development Kits (SDKs) -AWS provides SDKs for various programming languages, including Python, Java, JavaScript, .NET, and more. Developers can use these SDKs to integrate AWS services directly into their applications. This allows for programmatic access to AWS services, enabling applications to interact with AWS programmatically.

# ACCESSING AWS services

- AWS Management APIs - AWS exposes RESTful APIs for each of its services, allowing developers to programmatically interact with AWS services over HTTP. This provides a low-level interface for accessing AWS services and is commonly used when SDKs are not available in a specific programming language or when fine-grained control is required.
- AWS Mobile SDK- Specifically designed for mobile app developers, the AWS Mobile SDK allows developers to integrate AWS services into mobile applications. This SDK supports both iOS and Android platforms, enabling features such as authentication, data storage, and analytics within mobile apps.

# ACCESSING AWS services

- AWS CloudFormation - CloudFormation is AWS's infrastructure as code service, allowing users to define and provision AWS infrastructure using YAML or JSON templates. With CloudFormation, users can create and manage AWS resources in a repeatable and automated manner.
- AWS Management Console Mobile App - AWS offers a mobile app for iOS and Android devices, providing users with on-the-go access to their AWS resources and services. Users can view resource status, receive alerts, and perform basic management tasks from their mobile devices.

# PRACTICAL: AWS CLI

# INSTALL AWS CLI

- Haves
  1. Virtualization software : virtual box / VMWARE
  2. Kali linux or ubuntu
- STEPS
  1. UPDATE PACKAGES : sudo apt-get update
  2. INSTALL PIP : sudo apt-get install python3-pip
  3. INSTALL AWS CLI : sudo pip install awscli
  4. VERIFIY : aws –-version
  5. CONFIGURE : aws configure

# AWS S3 BUCKET

- Amazon Simple Storage Service (S3) is a scalable object storage service offered by Amazon Web Services (AWS).
- It provides developers and IT teams with secure, durable, and highly scalable storage for various purposes, such as data backup and recovery, application hosting, media storage, and data archiving.
- S3 stores data as objects within buckets. An object consists of data, metadata, and a unique identifier.
- A bucket is a container for objects stored in S3. Buckets have a globally unique name and can be configured with various settings, such as access control and versioning.

# AWS S3 BUCKET

- S3 can scale to accommodate virtually unlimited amounts of data. It automatically scales to handle growing storage needs without requiring any manual intervention.
- offers several security features to help protect data, including encryption at rest and in transit, access control lists (ACLs), and bucket policies. Users can also integrate S3 with AWS Identity and Access Management (IAM) to manage access to their buckets and objects..
- S3 supports versioning, which allows users to keep multiple versions of an object in the same bucket. This helps protect against accidental deletion or overwrites and provides a way to restore previous versions if needed.

# AWS S3 BUCKET

# AWS S3 BUCKET NAMING RULES

The following naming rules apply for general purpose buckets.

1. Bucket names must be between 3 (min) and 63 (max) characters long.
2. Bucket names can consist only of lowercase letters, numbers, dots (.), and hyphens (-).
3. Bucket names must begin and end with a letter or number
4. Bucket names must not contain two adjacent periods.
5. Bucket names must not be formatted as an IP address (for example, 192.168.5.4).

# AWS S3 BUCKET

1. Bucket names must not start with the prefix xn--.
2. Bucket names must not start with the prefix sthree- and the prefix sthree-configurator.
3. Bucket names must not end with the suffix -s3alias. This suffix is reserved for access point alias names. For more information, see Using a bucket-style alias for your S3 bucket access point.
4. Bucket names must not end with the suffix --ol-s3. This suffix is reserved for Object Lambda Access Point alias names. For more information, see How to use a bucket-style alias for your S3 bucket Object Lambda Access Point.

# AWS S3 BUCKET

1. Bucket names must be unique across all AWS accounts in all the AWS Regions within a partition. A partition is a grouping of Regions. AWS currently has three partitions: aws (Standard Regions), aws-cn (China Regions), and aws-us-gov (AWS GovCloud (US)).
2. A bucket name cannot be used by another AWS account in the same partition until the bucket is deleted.
3. Buckets used with Amazon S3 Transfer Acceleration can't have dots (.) in their names. For more information about Transfer Acceleration, see Configuring fast, secure file transfers using Amazon S3 Transfer Acceleration.

# AWS S3 BUCKET NAMING EXAMPLE

Example general purpose bucket names

The following example bucket names are valid and follow the recommended naming guidelines for general purpose buckets:

1. docexamplebucket1
2. log-delivery-march-2020
3. my-hosted-content

# AWS S3 BUCKET NAMING EXAMPLE

The following example bucket names are valid but not recommended for uses other than static website hosting:

1. docexamplewebsite.com
2. www.docexamplewebsite.com
3. my.example.s3.bucket

# AWS S3 BUCKET NAMING EXAMPLE

The following example bucket names are not valid:

1. doc_example_bucket (contains underscores)
2. DocExampleBucket (contains uppercase letters)
3. doc-example-bucket- (ends with a hyphen)

# HOSTING A STATIC WEBSITE WITH AWS S3

The following example bucket names are not valid:

1. doc_example_bucket (contains underscores)
2. DocExampleBucket (contains uppercase letters)
3. doc-example-bucket- (ends with a hyphen)

# LAB 1: Configuring a static website on Amazon S3

LAB SERIES

# STEP 1

## Create a bucket

NOTE;

Obey the naming rules

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose Create bucket.
3. Enter the Bucket name (for example, example.com).
4. Choose the Region where you want to create the bucket.
5. Choose a Region that is geographically close to you to minimize latency and costs, or to address regulatory requirements. The Region that you choose determines your Amazon S3 website endpoint. For more information, see Website endpoints.
6. To accept the default settings and create the bucket, choose Create.

# STEP 2

Enable static website hosting

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. In the Buckets list, choose the name of the bucket that you want to enable static website hosting for.
3. Choose Properties.
4. Under Static website hosting, choose Edit.
5. Choose Use this bucket to host a website.
6. Under Static website hosting, choose Enable.
7. In Index document, enter the file name of the index document, typically index.html.(case sensitive)
8. To provide your own custom error document for 4XX class errors, in Error document, enter the custom error document file name.

# STEP 2

## Enable static website hosting

1. Choose Save changes.
2. Amazon S3 enables static website hosting for your bucket. At the bottom of the page, under Static website hosting, you see the website endpoint for your bucket.
3. Under Static website hosting, note the Endpoint.
4. The Endpoint is the Amazon S3 website endpoint for your bucket. After you finish configuring your bucket as a static website, you can use this endpoint to test your website.

# HOSTING A STATIC WEBSITE WITH AWS S3

The following example bucket names are not valid:

1. doc_example_bucket (contains underscores)
2. DocExampleBucket (contains uppercase letters)
3. doc-example-bucket- (ends with a hyphen)

# LAB 2: Building a Serverless Web Application

with AWS Lambda, Amazon API Gateway, AWS Amplify, Amazon DynamoDB, and Amazon Cognito

# Aim & Objectives Goal

In this lab session ;

- The aim is to create a serverless web application that enables users to request unicorn rides from the  fleet service.
- The application will present users with an HTML-based user interface for indicating a pick-up location and a RESTful web service on the backend to submit the request for dispatching a unicorn.
- The application will also provide facilities for users to register with the service and log in before requesting rides.

# TOOLS / TECH

AWS ACCOUNT

GIT ACCOUNT

AWS CLI

GIT BASH

# Application architecture

- The application architecture uses;
- AWS Lambda, Amazon API Gateway, Amazon DynamoDB, Amazon Cognito, and AWS Amplify Console.

# Application architecture

- **AWS Lambda-** AWS Lambda is a serverless compute service that lets you run code without provisioning or managing servers. You can upload your code and Lambda takes care of everything required to run and scale your code with high availability.

# Application architecture

- **Amazon API Gateway** - Amazon API Gateway is a fully managed service that makes it easy for developers to create, publish, maintain, monitor, and secure APIs at any scale. It acts as a front door for applications to access data, business logic, or functionality from backend services.

# Application architecture

- **Amazon DynamoDB** - Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. It offers low-latency access to data at any scale and supports both document and key-value data models.

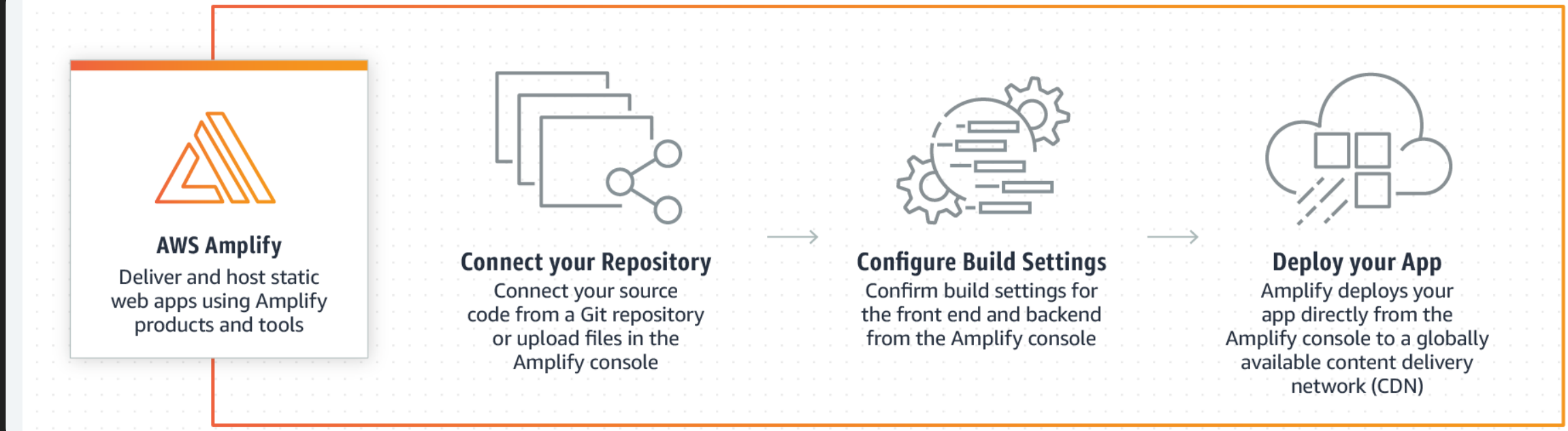
# Application architecture

- **Amazon Cognito** - Amazon Cognito is a fully managed identity service that provides authentication, authorization, and user management for web and mobile apps. It allows you to easily add user sign-up and sign-in to your apps, as well as manage user identities and access control.

# Application architecture

- **AWS Amplify Console** - AWS Amplify Console is a continuous deployment and hosting service for web applications with serverless backends. It automatically builds and deploys your frontend and backend code, and provides features like custom domains, SSL/TLS certificates, and Git-based workflows to streamline the development and deployment process.

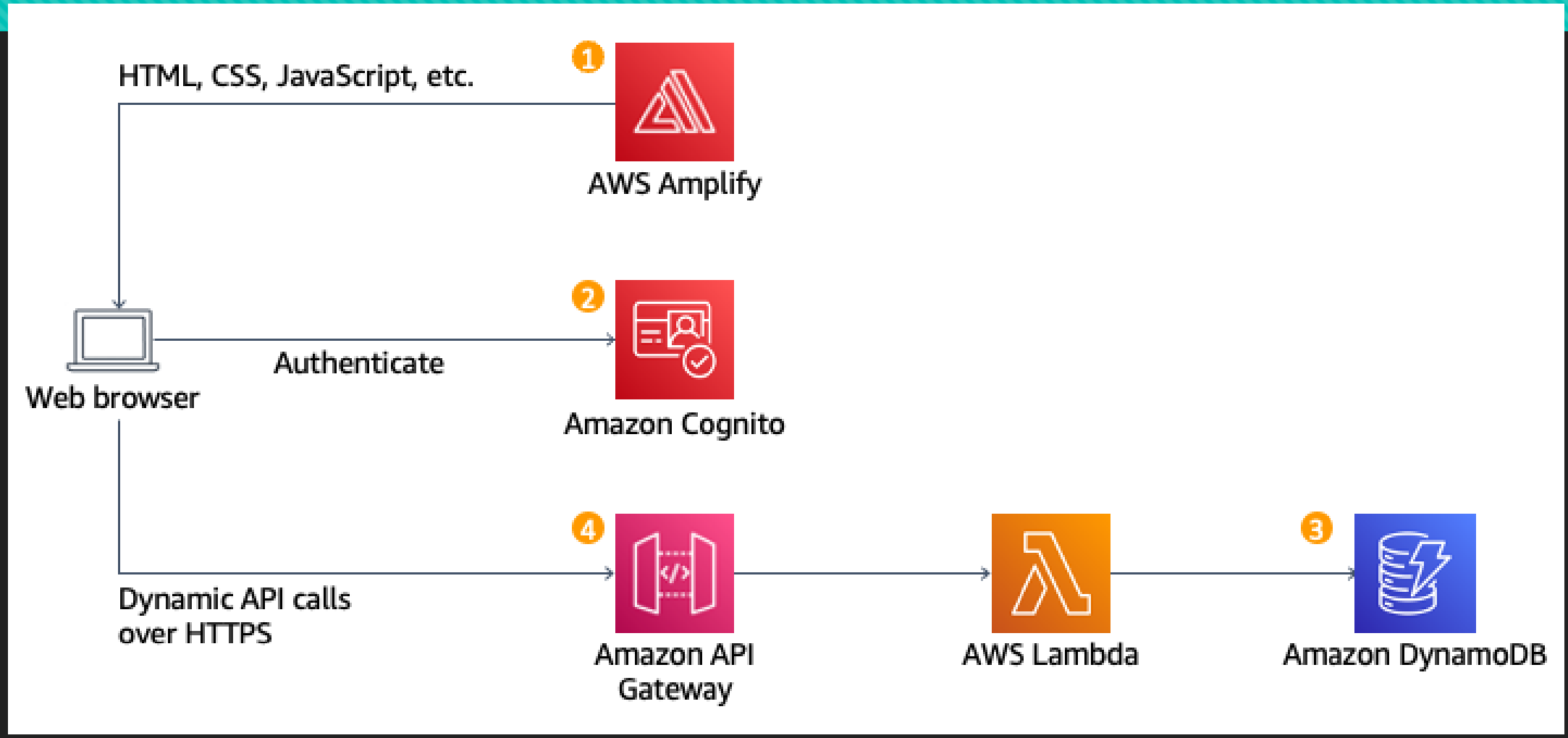
# Application architecture
## AWS Amplify Console



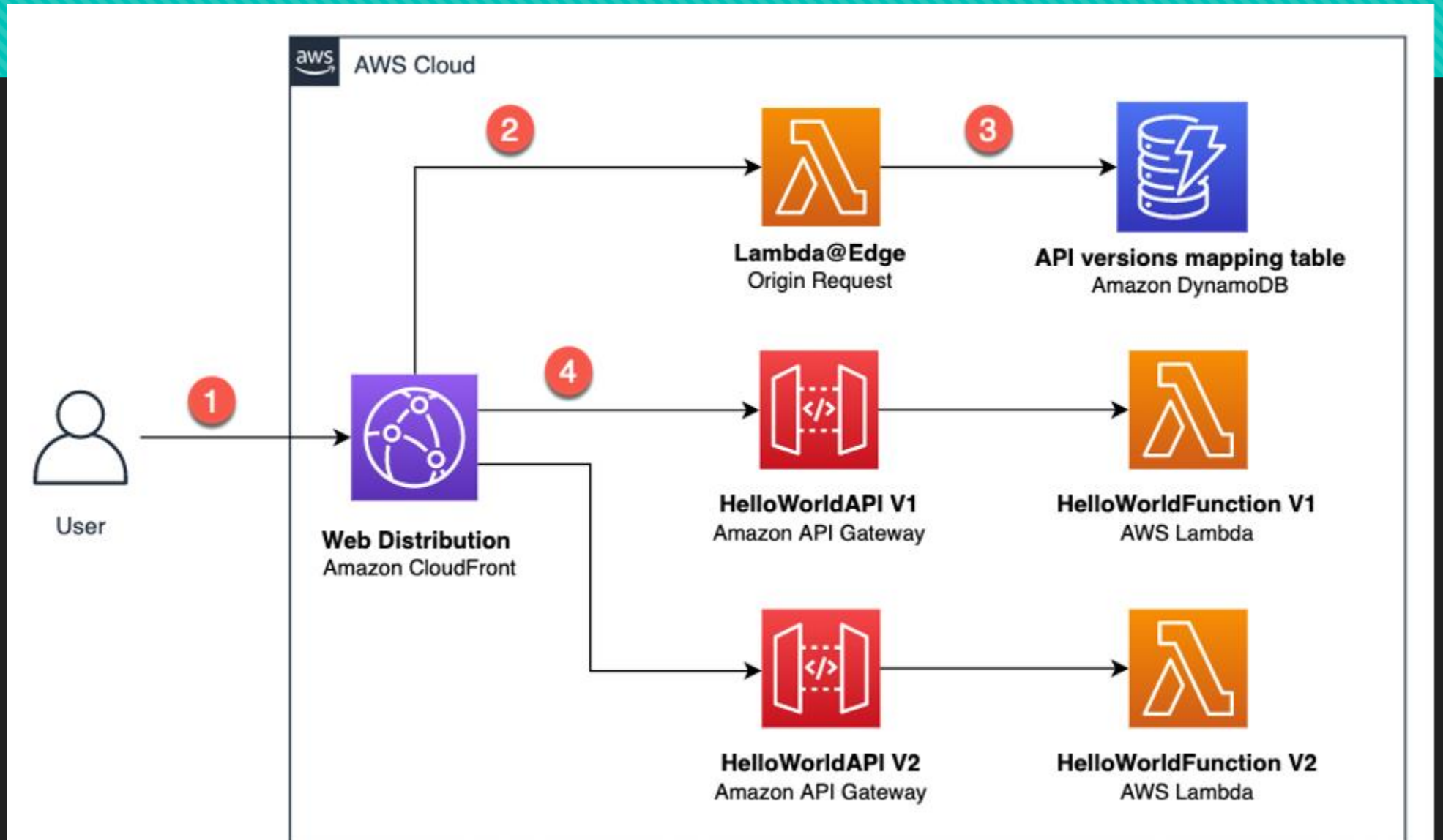
# Application architecture

- **Amplify Console** provides continuous deployment and hosting of the static web resources including HTML, CSS, JavaScript, and image files which are loaded in the user's browser.

# Application architecture



# Application architecture

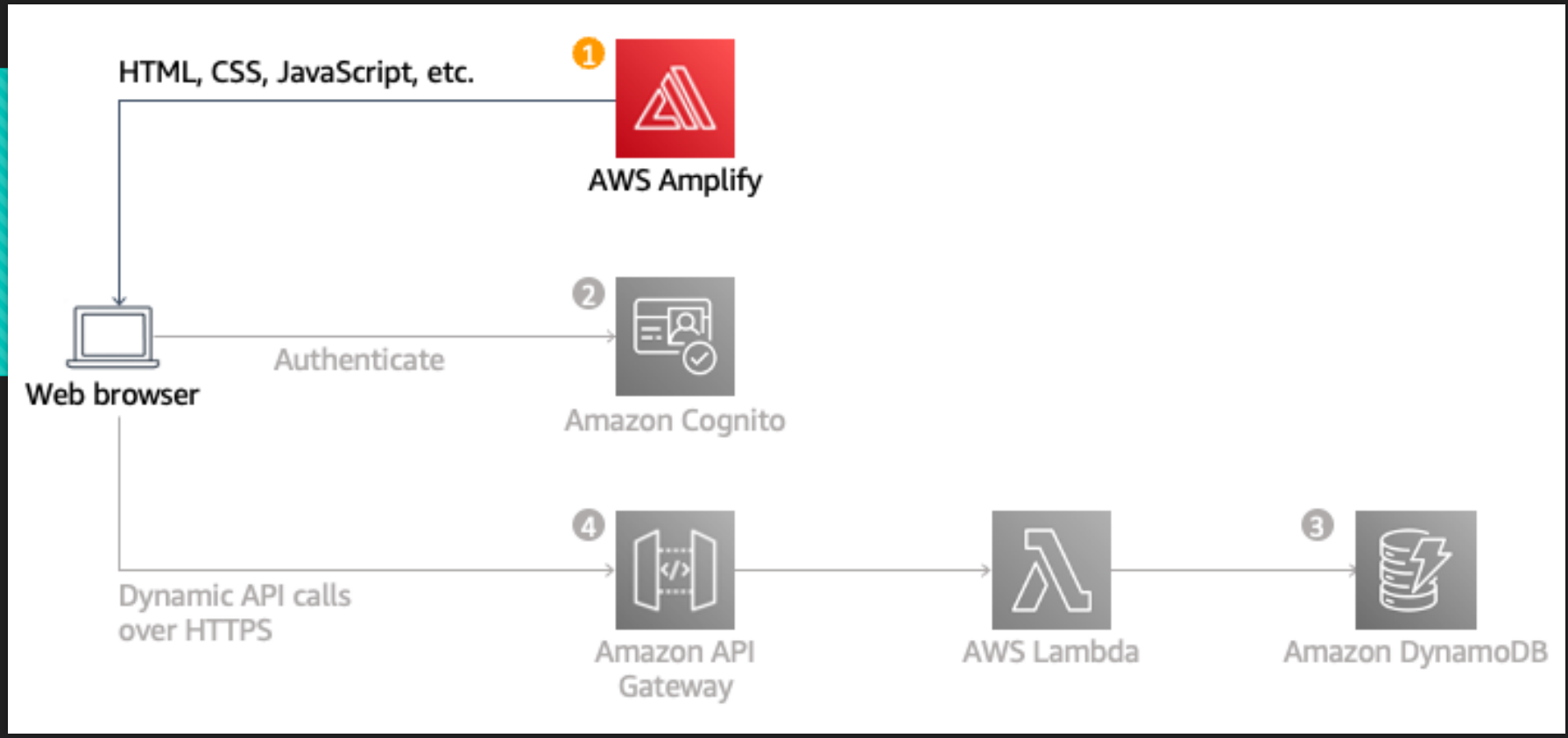


# Application architecture icons

# STEPS

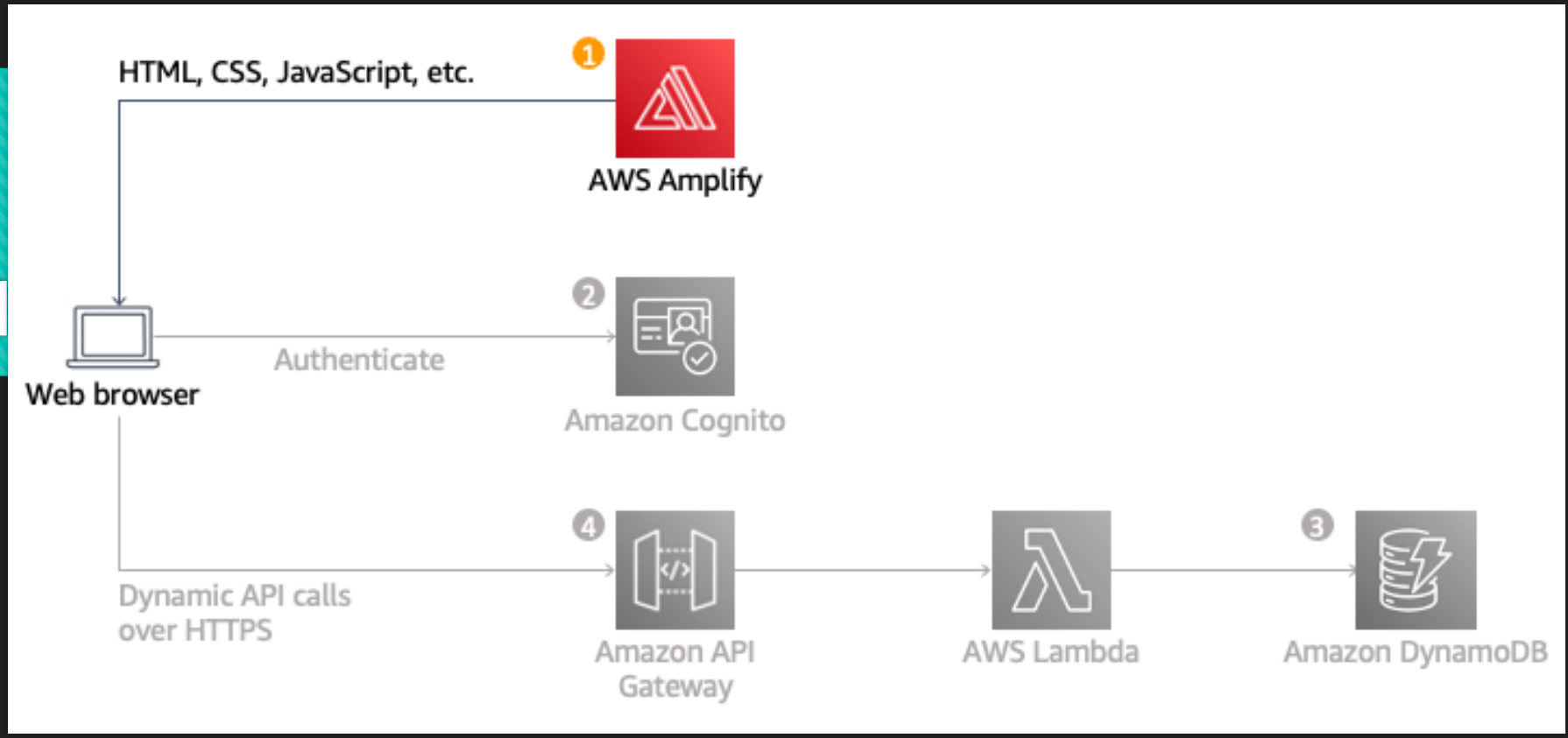
1. Host a Frontend app(static Website): Configure AWS Amplify to host the static resources for your web application with continuous deployment built in
2. Manage Users : Create an Amazon Cognito user pool to manage your users' accounts
3. Build a Serverless Backend : Build a backend process for handling requests for your web application
4. Deploy a RESTful API : Use Amazon API Gateway to expose the Lambda function you built in the previous module as a RESTful API
5. Terminate Resources : Terminate all the resources you created throughout this tutorial

# STEP 1



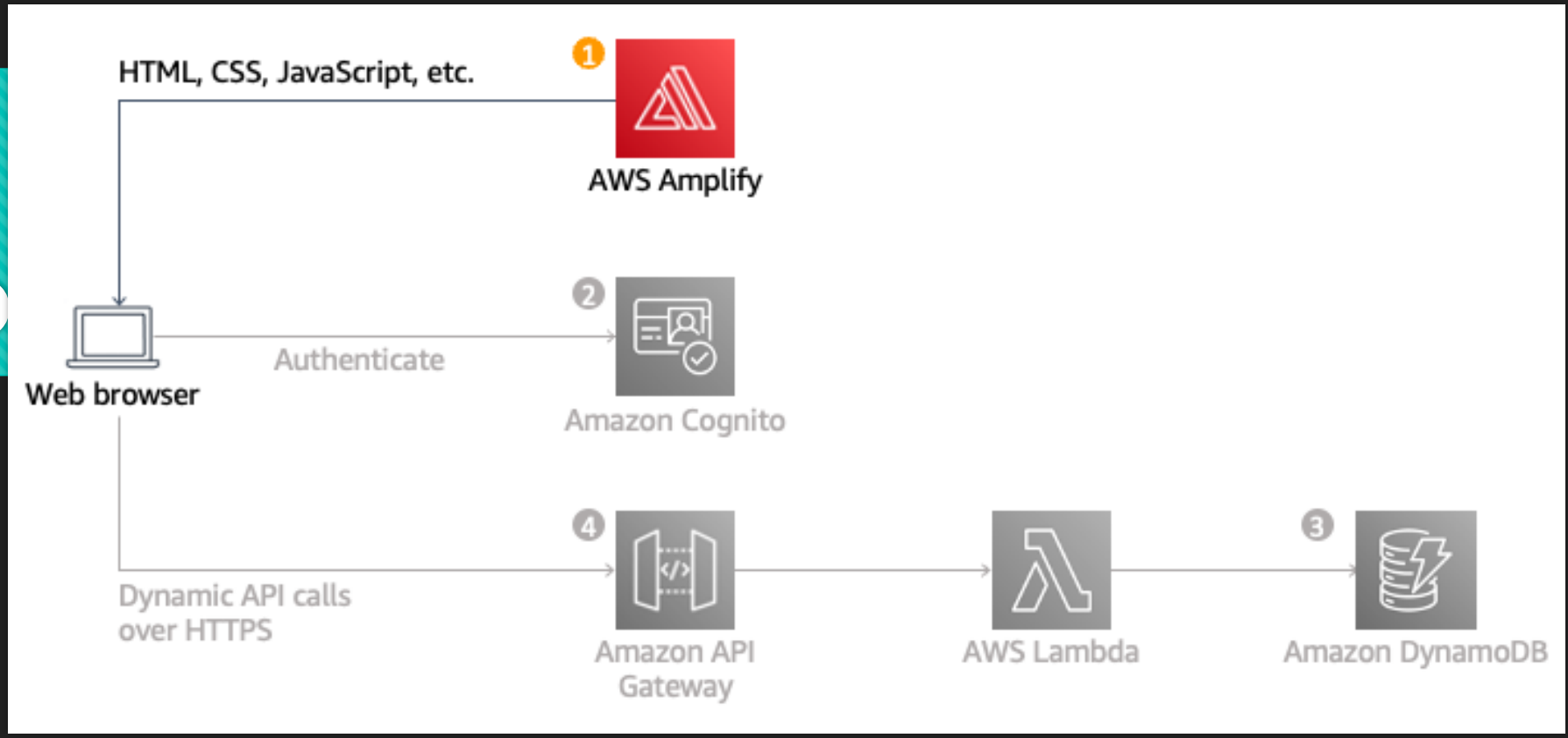

- Here we will configure AWS Amplify to host the static resources for the web application with continuous deployment built in.
- The Amplify Console provides a git-based workflow for continuous deployment and hosting of full-stack web app

# STEP 1a




- **SELECTING A REGION**
- **US East (N. Virginia)**

# STEP 1b




- CREATE A GIT REPOSITORY
- search for AWS codecommit We will use it to store our application code, but you can do the same by creating a repository on GitHub.
- Install AWS CLI

# STEP 1c

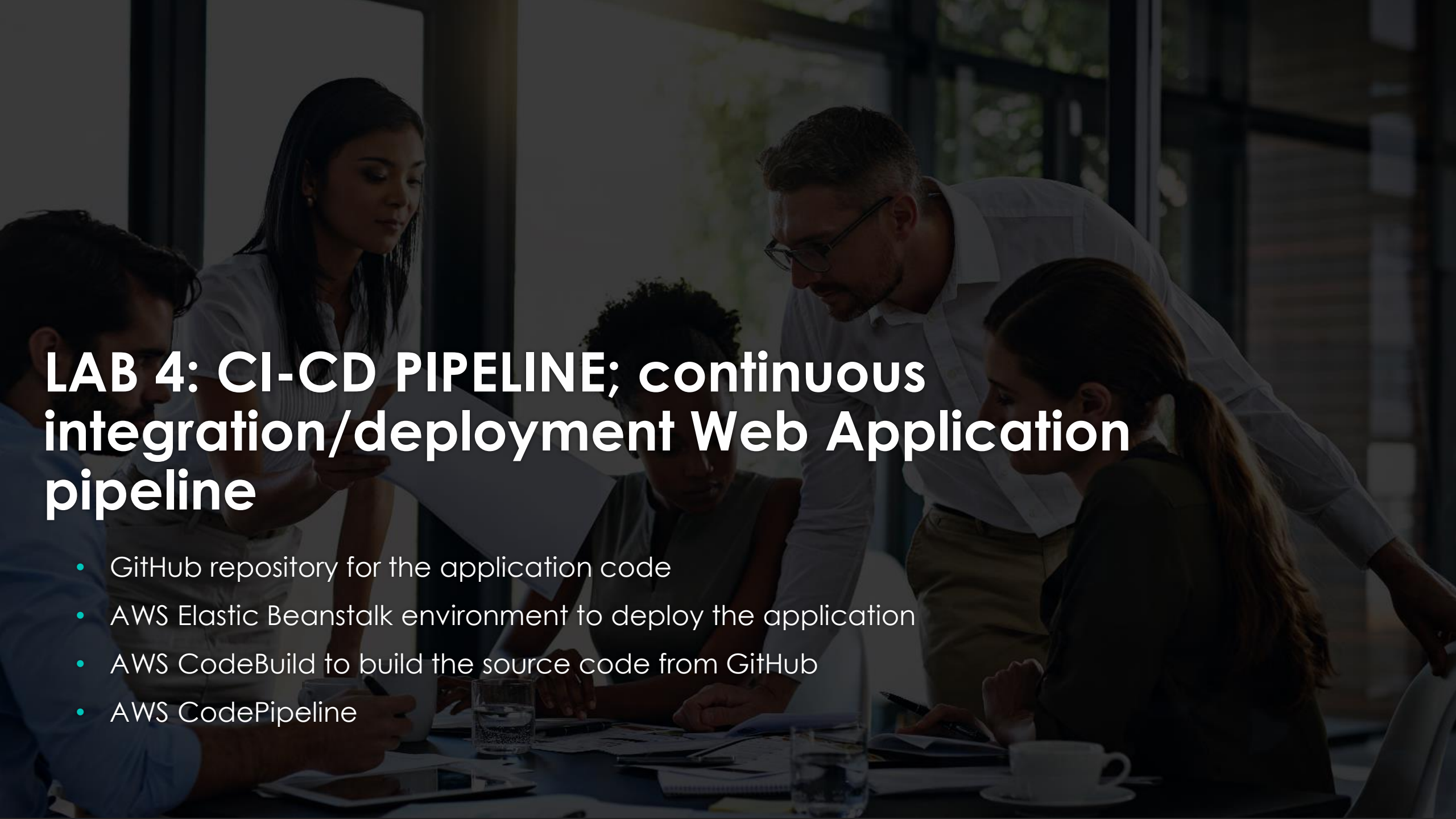
- CREATE A GIT REPOSITORY set up an IAM user with Git credentials in the IAM console
- creating IAM users (console)
- On the Console Home page, select the IAM service.
- In the navigation pane, select Users and then select Add users.
- On the Specify user details page, under User details, in User name, enter the name for the new user. This is their sign-in name for AWS

# STEP 1d

- Change directory into your repository and copy the static files from S3 using the following commands (make sure you change the Region in the following command to copy the files from the S3 bucket to the Region you selected at the beginning of this lab:
- cd wildrydes-site
- aws s3 cp s3://wildrydes-us-east-1/WebApplication/1_StaticWebHosting/website ./ --recursive

# STEP 1d

- Add, commit, and push the git files.
- The following code block is an example of what you will see in your terminal window:
- $ git add .
- $ git commit -m "new files"
- $ git push



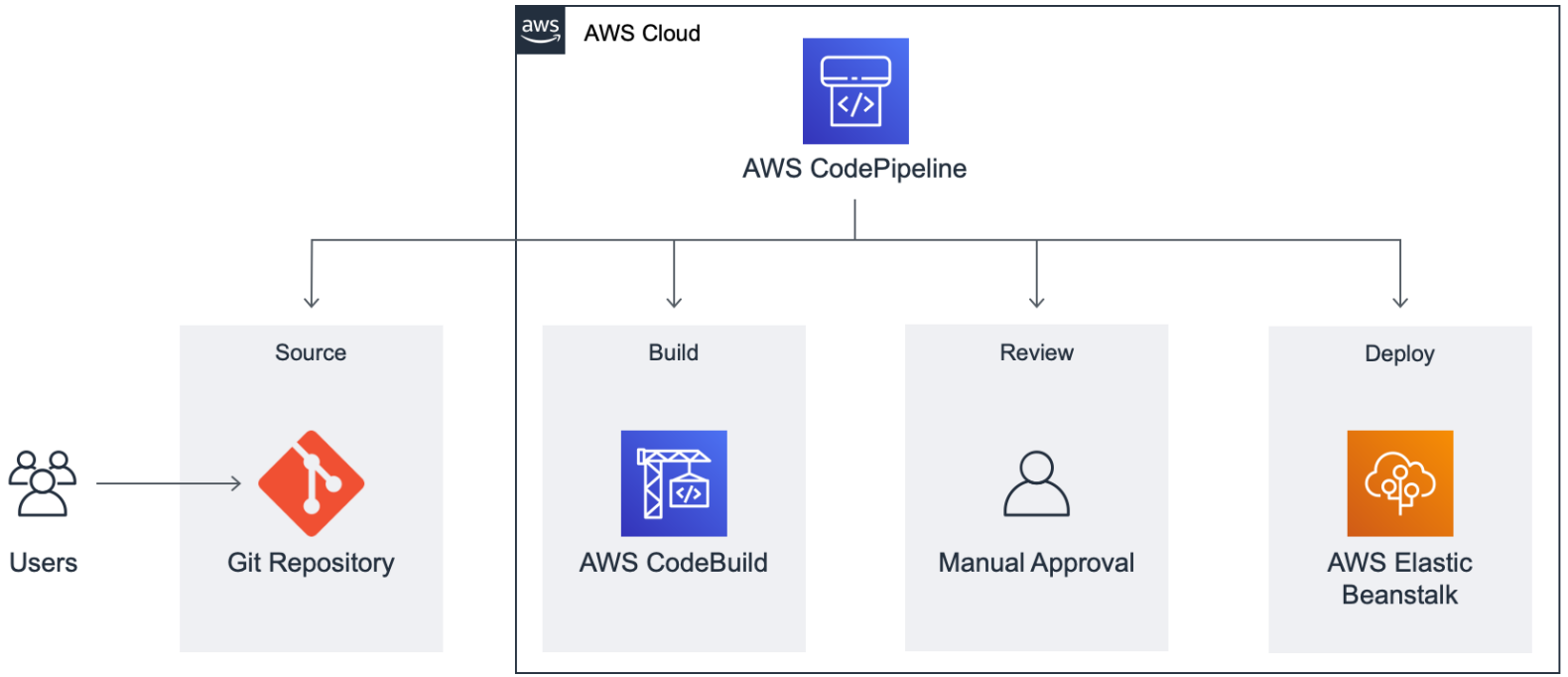
# LAB 4: CI-CD PIPELINE; continuous integration/deployment Web Application pipeline

- GitHub repository for the application code
- AWS Elastic Beanstalk environment to deploy the application
- AWS CodeBuild to build the source code from GitHub
- AWS CodePipeline

# AWS cloud services Needed

- AWS Elastic Beanstalk - A service that makes it easy to deploy your application on AWS. You simply upload your code and Elastic Beanstalk deploys, manages, and scales your application.
- Environment - Collection of AWS resources provisioned by Elastic Beanstalk that are used to run your application.
- EC2 instance - Virtual server in the cloud. Elastic Beanstalk will provision one or more Amazon EC2 instances when creating an environment.
- Web server - Software that uses the HTTP protocol to serve content over the Internet. It is used to store, process, and deliver web pages.
- Platform—Combination of operating system, programming language runtime, web server, application server, and Elastic Beanstalk components. Your application runs using the components provided by a platform.

# The app AWS cloud Architecture



# STEP 1 - Git

- Given Git on your PC, log into your account.
- In that same tab, open the CSC314-CLOUD-COMP repo.
- Choose the white Fork button on the top right corner.
- Verify it is showing your account and choose Create a fork. After a few seconds, your browser will display a copy of the repo in your account under Repositories.

# STEP 1a

1. Go to the repository and choose the green Code button near the top of the page.
2. To clone the repository using HTTPS, confirm that the heading says *Clone with HTTPS.* If not, select the Use HTTPS link.
3. Choose the white button with a clipboard icon on it (to the right of the URL).

# STEP 2
# DEPLOYMENT

## AWS Elastic Beanstalk

- In a new browser tab, open the AWS Elastic Beanstalk console.
- Choose the orange Create Application button.
- Choose Web server environment under the Configure environment heading.
- In the text box under the heading Application name, enter **CSC314DevOPs.**

# STEP 2
# DEPLOYMENT

- In the Platform dropdown menu, under the Platform heading, select Node.js . Platform branch and Platform version will automatically populate with default selections.
- Confirm that the radio button next to Sample application under the Application code heading is selected.
- Confirm that the radio button next to Single instance (free tier eligible) under the Presets heading is selected.
- Select Next.

# STEP 3
# DEPLOYMENT

- Create a build project with AWS CodeBuild
- Set up GitHub as the source provider for a build project
- Run a build on AWS CodeBuild

# STEP 3 DEPLOYMENT

KEYNOTE SERVICES FOR THIS STEP

- Build process—Process that converts source code files into an executable software artifact. It may include the following steps: compiling source code, running tests, and packaging software for deployment.
- Continuous integration—Software development practice of regularly pushing changes to a hosted repository, after which automated builds and tests are run.
- Build environment—Represents a combination of the operating system, programming language runtime, and tools that CodeBuild uses to run a build.

# STEP 3 DEPLOYMENT

KEYNOTE SERVICES FOR THIS STEP

- Buildspec—Collection of build commands and related settings, in YAML format, that CodeBuild uses to run a build.
- Build Project—Includes information about how to run a build, including where to get the source code, which build environment to use, which build commands to run, and where to store the build output.
- OAuth—Open protocol for secure authorization. OAuth enables you to connect your GitHub account to third-party applications, including AWS CodeBuild.

# STEP 3 DEPLOYMENT

KEYNOTE SERVICES ON THIS STEP

- CHECK THE LAB NOTE FOR CONTINUATION OF THE STEPS