CHAPTER 1:

What is Cloud Computing?

Cloud computing is the on-demand delivery of IT resources over the Internet with pay-as-you-go pricing. OR Computing and software resources that are delivered on demand, as service..

What is Grid Computing?Grid Computing can be defined as a network of homogenous or heterogeneous computers working together over a long distance to perform a task that would rather be difficult for a single machine.

Common characteristics of Cloud Computing:

On-demand self-service: On-demand self-service in cloud computing means you can get the computer resources you need, like storage or processing power, whenever you want, without requiring human interaction.

Broad network access: Broad network access in cloud computing means you can use your stuff stored on the cloud from anywhere with the internet, using different devices like your laptop, phone, or tablet. It's like checking your email from any computer – you don't have to be at a specific place to access your things.

Resource pooling: Resource pooling in cloud computing means that the available computer resources, like storage and processing power, are shared and used efficiently among many users. It's like a community pool where everyone can use the water without needing to have their own private pool.

Rapid elasticity: Rapid elasticity in cloud computing is like a stretchy rubber band. You can quickly and easily get more computer power or storage space when you need it, and then shrink it back down when you're done. It's flexible and adjusts to your needs just like a rubber band stretches and contracts.

Measured service / Flexible pricing - Pay per use : Measured service in cloud computing is like paying for your electricity or water usage. You only pay for what you actually use – no more, no less. Similarly, in the cloud, you're charged for the resources you use, such as storage or processing, so it's fair and cost-efficient.

Resiliency: Resiliency in cloud computing is like having a backup plan. If one part of the cloud has a problem, your stuff is still safe and available because it's stored in different places too.

Advantages of Cloud Computing

Cost: It reduces the huge capital costs of buying hardware and software.

Speed: Resources can be accessed in minutes, typically within a few clicks.

Scalability: We can increase or decrease the requirement of resources according to the business requirements.

Productivity: While using cloud computing, we put less operational effort. We do not need to apply patching, as well as no need to maintain hardware and software. So, in this way, the IT team can be more productive and focus on achieving business goals.

Reliability: Backup and recovery of data are less expensive and very fast for business continuity.

Security: Many cloud vendors offer a broad set of policies, technologies, and controls that strengthen our data security.

Unlimited Storage Capacity.

Device Independence and the “always on!, anywhere and any place”

Cloud Service models/Cloud delivery models

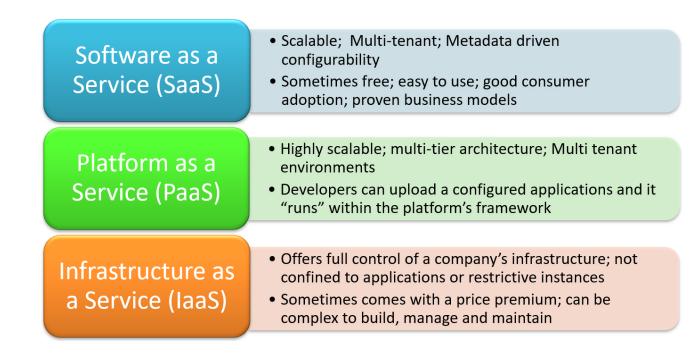
Software-as-a-Service (SaaS): https://youtu.be/vio29hRGMFI

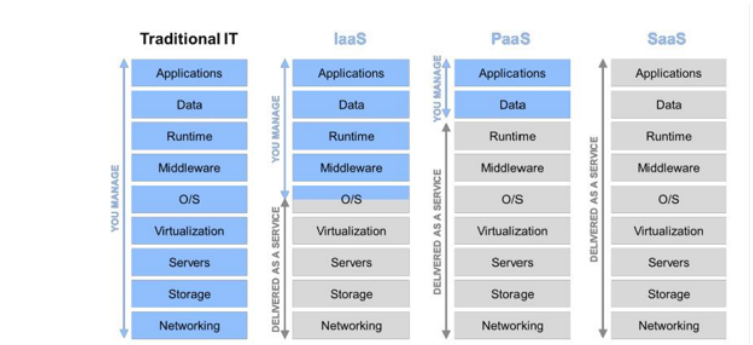
Platform-as-a-Service (PaaS): https://youtu.be/M8e9XSIIS3E

Infrastructure-as-a-Service (IaaS): https://youtu.be/VLCYXJ-Xvy8

IaaS is also known as Hardware as a Service (HaaS). It is a computing infrastructure managed over the internet. The main advantage of using IaaS is that it helps users to avoid the cost and complexity of purchasing and managing the physical servers.

Difference Between IAAS , PAAS , SAAS: https://www.geeksforgeeks.org/difference-between-iaas-paas-and-saas/





Cloud implementation types (Deployment Models): https://youtu.be/jiHcbUr42zg

Comparison of Top Cloud Deployment Models

	Public	Private	Community	Hybrid
Ease of setup and use	Easy	Requires IT proficiency	Requires IT proficiency	Requires IT proficiency
Data security and privacy	Low	High	Comparatively high	High
Data control	Little to none	High	Comparatively high	Comparatively high
Reliability	Low	High	Comparatively high	High
Scalability and flexibility	High	High	Fixed capacity	High
Cost-effectiveness	The cheapest	Cost-intensive; the most expensive model	Cost is shared among community members	Cheaper than a private model but more costly than a public one
Demand for in-house hardware	No	Depends	Depends	Depends

Disadvantages of Cloud Computing:

Requires good speed internet with good bandwidth: To access your cloud services, you need to have a good internet connection always with good bandwidth to upload or download files to/from the cloud

Downtime: Since the cloud requires high internet speed and good bandwidth, there is always a possibility of service outage, which can result in business downtime. Today, no business can afford revenue or business loss due to downtime or slow down from an interruption in critical business processes.

Limited control of infrastructure: Since you are not the owner of the infrastructure of the cloud, hence you don't have any control or have limited access to the cloud infra.

Restricted or limited flexibility: The cloud provides a huge list of services, but consuming them comes with a lot of restrictions and limited flexibility for your applications or developments. Also, platform dependency or 'vendor lock-in' can sometimes make it difficult for you to migrate from one provider to another.

Ongoing costs: Although you save your cost of spending on whole infrastructure and its management, on the cloud, you need to keep paying for services as long as you use them. But in traditional methods, you only need to invest once.

Security: Security of data is a big concern for everyone. Since the public cloud utilizes the internet, your data may become vulnerable. In the case of a public cloud, it depends on the cloud provider to take care of your data. So, before opting for cloud services, it is required that you find a provider who follows maximum compliance policies for data security.

Vendor Lock-in: Although the cloud service providers assure you that they will allow you to switch or migrate to any other service provider whenever you want, it is a very difficult process. You will find it complex to migrate all the cloud services from one service provider to another. During migration, you might end up facing compatibility, interoperability and support issues. To avoid these issues, many customers choose not to change the vendor.

Technical issues: Even if you are a tech whiz, the technical issues can occur, and everything can't be resolved in-house. To avoid interruptions, you will need to contact your service provider for support. However, not every vendor provides 24/7 support to their clients.

Cluster computing: Cluster computing is a collection of tightly or loosely connected computers that work together so that they act as a single entity. The connected computers execute operations all together thus creating the idea of a single system. The clusters are generally connected through fast local area networks (LANs)

Cloud Computing Architecture: https://youtu.be/2Dd2ducs-ic

Grid Computing vs Cloud Computing Difference:

S.NO	Cloud Computing	Grid Computing
1.	Cloud computing is a Client-server computing architecture.	While it is a Distributed computing architecture.
2.	Cloud computing is a centralized executive.	While grid computing is a decentralized executive.
3.	In cloud computing, resources are used in centralized pattern.	While in grid computing, resources are used in collaborative pattern.
4.	It is more flexible than grid computing.	While it is less flexible than cloud computing.
5.	In cloud computing, the users pay for the use.	While in grid computing, the users do not pay for use.
6.	Cloud computing is a high accessible service.	While grid computing is a low accessible service.
7.	It is highly scalable as compared to grid computing.	While grid computing is low scalable in comparison to cloud computing.
8.	It can be accessed through standard web protocols.	While it is accessible through grid middleware.
9.	Cloud computing is based on service-oriented.	Grid computing is based on application-oriented.
10.	Cloud computing uses service like IAAS, PAAS, SAAS.	Grid computing uses service like distributed computing, distributed pervasive, distributed information.

Cluster Computing vs Grid Computing:

CLUSTER COMPUTING	GRID COMPUTING
Nodes must be homogenous i.e. they should have same type of hardware and operating system.	Nodes may have different Operating systems and hardwares. Machines can be homogenous or heterogenous.
Computers in a cluster are dedicated to the same work and perform no other task.	Computers in a grid contribute their unused processing resources to the grid computing network.
Computers are located close to each other.	Computers may be located at a huge distance from one another.
Computers are connected by a high speed local area network bus.	Computers are connected using a low speed bus or the internet.
Computers are connected in a centralized network topology.	Computers are connected in a distributed or de-centralized network topology.
Scheduling is controlled by a central server.	It may have servers, but mostly each node behaves independently.
Whole system has a centralized resource manager.	Every node manages it's resources independently.
Whole system functions as a single system.	Every node is autonomous, and anyone can opt out anytime.

CHAPTER 2:

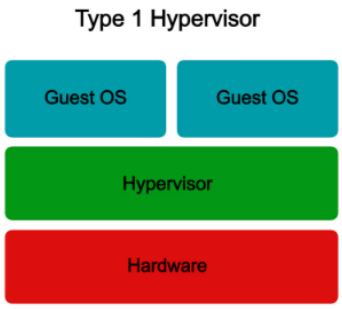
Virtualization Technology: https://youtu.be/7yXlLgZkiJM
https://youtu.be/2rwUGh6GDjM

Hypervisor: Basically it is a software that create, run and manage Virtual machines and It is also called VMM(Virtual machine monitor)

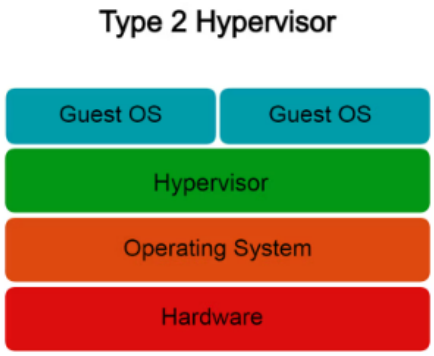
Types of Hypervisors: https://youtu.be/qmPCBBBrc00

Type 1 Hypervisors/ Native Hypervisors / Bare metal Hypervisor: https://youtu.be/ojpYJTz9SDg

hypervisors run directly on the system hardware – A "bare metal" embedded hypervisor. Examples are: 1) VMware ESX and ESXi 2) Microsoft Hyper-V 3) Citrix XenServer 4) Oracle VM



Type 2 Hypervisors/ Hosted Hypervisors / Embedded Hypervisors: https://youtu.be/n0uY8nvOvAM

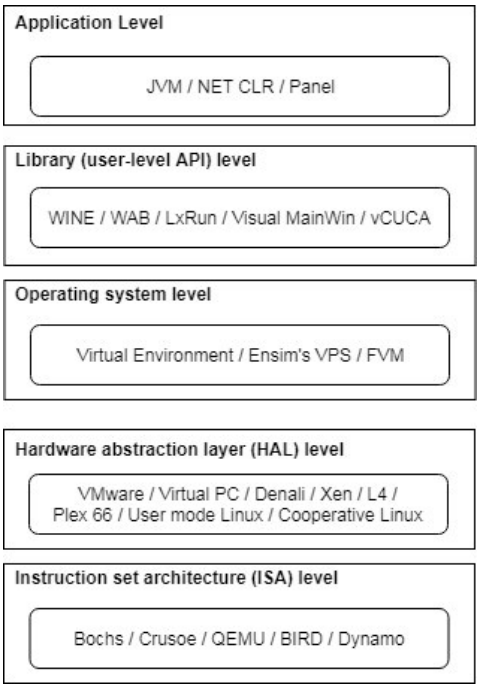


hypervisors run on a host operating system that provides virtualization services, such as I/O device support and memory management. Examples are: 1) VMware Workstation/Fusion/Player 2) Microsoft Virtual PC 3) Oracle VM VirtualBox 4) Red Hat Enterprise Virtualization

Category	Type 1	Type 2
Location Installed	Directly installed on computer hardware	Installed on top of the host OS
Virtualization Type	Hardware virtualization	OS virtualization
Operation	Guest OS and application on the hypervisor	As an application on OS
Performance	Takes advantage of high-core count processors more efficiently, making it ideal for big and high-scaling operations	Adequate for testing, development, and tinkering
Security	Direct hardware installation means each VM is very safe from all host OS vulnerabilities	Provides sandboxed guest OS making it adequately safe
Setup	Easy but some technical knowledge required	Quick and easy
Suited Hardware	Type 1 hypervisors get their performance from high processor core counts; server-rated hardware is ideal	Type 2 hypervisors are used for smaller-scale operations and convenience; better suited to PC hardware

CPU Virtualization: CPU virtualization, a key aspect of virtualization, involves creating virtual instances of central processing units (CPUs) within virtual machines. This allows multiple virtual machines to run on a single physical CPU, with each VM thinking it has its own dedicated CPU. CPU virtualization is achieved through a hypervisor, which manages the allocation and execution of virtual CPUs (vCPUs) on the physical CPU cores.

Implementation Levels of Virtualization:

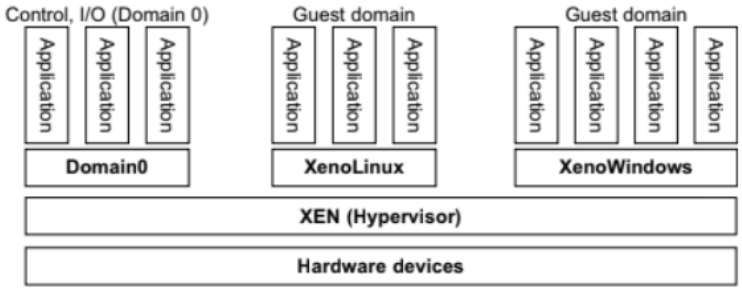


1): ISA (Instruction Set Architecture) level: ISA (Instruction Set Architecture) Virtualization is like having a translator that helps different types of computers understand and follow the same rules. In cloud computing, ISA virtualization allows diverse computer hardware to work together by translating their unique instructions into a common language that they all understand. This helps them cooperate and perform tasks smoothly in the cloud environment.
2): HAL (Hardware Abstraction Layer) Virtualization Level: In cloud computing, HAL virtualization ensures that different types of hardware can work harmoniously by providing a standardized interface that bridges the gaps between them. It's like using these connectors to bring together different hardware components so they can work together smoothly in the cloud.
3): Operating System Level Virtualization (Containerization): In cloud computing, containerization allows multiple applications to run on the same operating system, with each application having its own isolated environment while using shared system resources. It's akin to having different containers inside a single operating system, which promotes efficient resource utilization and isolation for applications in the cloud.
4): Library Level Virtualization: In cloud computing, library level virtualization provides a common set of tools, functions, and resources that multiple applications can use. This promotes efficiency because applications don't need to carry all their tools separately – they can access the shared toolbox. This level of virtualization makes it easier for applications to work together and collaborate effectively in the cloud environment.
5): Application Level Virtualization: In cloud computing, application level virtualization bundles together all the components that an application needs to run smoothly. This includes the software itself, along with its dependencies, settings, and configurations. everything required to make the application work is neatly packaged, making it easier to deploy and manage applications in the cloud without having to worry about gathering all the individual parts separately.

Virtualization Structures/Tools and Mechanisms:

<u>XEN Hypervisor:</u>

- XEN is an open source hypervisor program developed by Cambridge university.
- Xen is a microkernel hypervisor.
- Xen does not include any device drives natively.
- It provides a mechanism by which a guest os can have direct access to the physical devices.
- The core components of a Xen System are the hypervisor , kernel and applications



This Domain 0 virtual machine is directly connected with hardware and this guest domain virtual machines do not have direct access to hardware, domain 0 provides a way for these guest domains to access hardwares.
Here one disadvantage is that: Security Concern: If any hacker hacks Domain 0 then Hacker can hack the whole system.

CHAPTER 3:

All architecture in Detail: https://drive.google.com/file/d/10Xy8mage_Vo26Zj68NtYcKdcyglS5j2b/view?usp=sharing

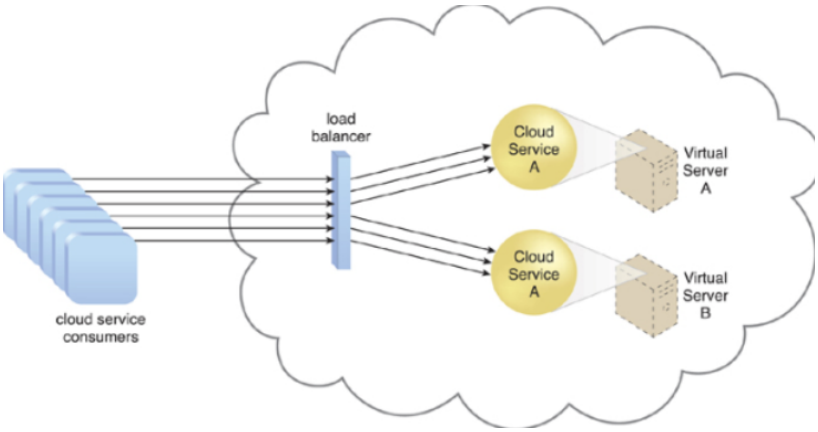
Workload Distribution Architecture:

Workload Distribution Architecture in cloud computing refers to the strategy of assigning tasks and computational workloads across multiple resources like servers or virtual machines within a cloud environment. This is done to achieve efficient resource utilization, better performance, and responsiveness.

Imagine a busy restaurant kitchen. Instead of having one chef do all the cooking, the workload is distributed among different chefs, each specializing in a particular type of cuisine or task. This ensures that orders are prepared faster and the kitchen runs smoothly.

In the cloud, there are different components and mechanisms that make workload distribution possible:

1. Load Balancing: This is like a traffic controller at a busy intersection. Load balancing ensures that incoming tasks and requests are evenly distributed among available resources. It prevents overload on a single server while making the best use of all resources.
2. Virtualization: Think of this as dividing a big cake into smaller slices. Virtualization allows multiple virtual machines (VMs) to run on a single physical server. Workloads are distributed across these VMs, ensuring efficient use of the server's capacity.
3. Auto Scaling: Auto scaling is similar to adjusting the number of seats in a movie theater based on the audience. Cloud resources automatically scale up or down in response to changes in workload. If demand increases, more resources are allocated; if demand decreases, excess resources are released.
4. Content Delivery Networks (CDNs): CDNs work like having multiple warehouses in different cities. Content is stored in various locations, and when someone requests it, the content is delivered from the nearest location. This reduces latency and ensures faster access.
5. Microservices Architecture: Microservices architecture breaks down applications into smaller, manageable components. Each component can be hosted on different resources, allowing for efficient distribution of workloads.



For example, in a web application hosted on the cloud, when many users simultaneously access the website, the workload distribution architecture might allocate different requests to different servers to handle the traffic efficiently.

Workload Distribution Architecture: This architecture focuses on evenly distributing tasks and processes across multiple computing resources. The goal is to optimize resource utilization, ensure efficient task completion, and prevent overloading of any single resource. It helps in achieving better performance and responsiveness by avoiding resource bottlenecks.

Use Case: An IT company is running a large-scale e-commerce platform. During peak shopping seasons, such as Black Friday, the website experiences a surge in traffic. To ensure smooth customer experiences and prevent server overload, the company implements a workload distribution architecture. This architecture evenly distributes user requests across multiple servers, preventing bottlenecks and maintaining responsive website performance.

Resource Pooling Architecture: In this architecture, computing resources such as processing power, memory, and storage are pooled together into a shared resource pool. This pool can be dynamically allocated and reallocated to different tasks and users based on their requirements. This approach promotes efficient resource utilization and flexibility in managing varying workloads.

Use Case: An IT company offers cloud-based software solutions to various clients. Instead of allocating dedicated hardware to each client, the company pools its resources, including processing power and storage, into a shared pool. This way, resources can be allocated dynamically to different clients based on their needs, optimizing resource utilization and reducing costs.

Dynamic Scalability Architecture: This architecture enables a system to automatically adjust its resources based on the current demand. It allows for seamless expansion or contraction of resources as needed, ensuring that the system can

handle varying workloads without manual intervention. This improves performance during peak times and reduces costs during periods of lower demand.

Use Case: A software-as-a-service (SaaS) provider offers project management tools. Users typically access the platform during business hours, leading to varying workloads. To ensure optimal performance without manual intervention, the provider employs dynamic scalability. The architecture automatically adds server capacity during peak hours and scales down during off-peak times, ensuring consistent service quality.

Elastic Resource Capacity Architecture: Elastic resource capacity architecture is about the system's ability to quickly and easily increase or decrease its resources based on demand. It ensures that the system can handle sudden spikes in workload without performance degradation, and resources can be scaled down when demand decreases to save costs.

Use Case: An IT company hosts a video streaming platform that experiences fluctuating demand due to content releases and live events. To accommodate sudden spikes in viewership, the company utilizes elastic resource capacity architecture. During popular events, the system automatically provisions additional servers and bandwidth to handle the increased load, maintaining smooth streaming experiences.

Service Load Balancing Architecture: This architecture involves distributing incoming network traffic across multiple servers or resources to prevent any single resource from being overwhelmed. It enhances system reliability, optimizes resource utilization, and ensures efficient handling of user requests.

Use Case: A cloud hosting company manages multiple websites for various clients. To prevent any single server from being overwhelmed and ensure high availability, the company employs service load balancing architecture. Incoming web traffic is evenly distributed among multiple servers, ensuring that no single server becomes a performance bottleneck.

Cloud Bursting Architecture: Cloud bursting architecture refers to the ability of a system to dynamically expand its resources into a cloud environment during periods of high demand. This allows the system to handle sudden surges in workload without impacting performance, and then scale back down when the demand subsides.

Use Case: An IT company hosts a web application that is used primarily by employees during working hours. However, the company occasionally runs marketing campaigns that attract a much larger audience. To handle these traffic spikes, the company employs cloud bursting architecture. During campaign periods, the application dynamically scales up by utilizing additional cloud resources.

Elastic Disk Provisioning Architecture: This architecture involves dynamically allocating storage space to virtual machines or applications based on their needs. It ensures that storage resources can be adjusted as needed, preventing wastage and promoting efficient utilization.

Use Case: A data analytics company collects and processes vast amounts of data from various sources. Storage requirements can vary significantly depending on the volume of data being processed. To efficiently manage storage needs, the company uses elastic disk provisioning architecture. Storage resources are allocated and deallocated based on data processing requirements, ensuring cost-effective storage utilization.

Redundant Storage Architecture: Redundant storage architecture focuses on providing data redundancy and fault tolerance. It involves duplicating data across multiple storage devices to ensure that data remains accessible even if one storage device fails. This enhances data reliability and availability.

Use Case: A financial services company relies on a database to store sensitive customer data. To ensure data integrity and availability, the company implements redundant storage architecture. Data is replicated across multiple storage devices, ensuring that even if one device fails, data remains accessible from other devices, minimizing business disruptions.

Hypervisor Clustering Architecture: In this architecture, multiple hypervisors (virtualization platforms) are clustered together to provide high availability and fault tolerance for virtualized environments. If one hypervisor fails, the workload is automatically transferred to another hypervisor, minimizing downtime.

Use Case: An IT company hosts virtualized environments for its clients' applications. To ensure high availability and minimize downtime, the company employs hypervisor clustering architecture. If a hardware failure occurs on one hypervisor, the workload automatically shifts to another hypervisor in the cluster, ensuring continuous service availability.

Load Balanced Virtual Server Instances Architecture: This architecture involves distributing virtual server instances across multiple physical servers to evenly distribute the processing load. Load balancing ensures that no single server becomes overwhelmed, leading to improved performance and resource utilization.

Use Case: A gaming company hosts multiplayer online games that require real-time interaction between players. To ensure low latency and even distribution of players, the company employs load balanced virtual server instances architecture. Player connections are distributed across multiple servers, preventing server overload and providing a seamless gaming experience.

CHAPTER 4:

Edge computing purpose and definition:

Cloud computing comes with a number of drawbacks. The biggest problem of cloud computing is latency because of the distance between users and the data centers that host the cloud services. This has led to the development of a new technology called edge computing moves computing closer to end users.

Edge computing is a distributed IT architecture which moves computing resources from clouds and data centers as close as possible to the originating source. The main goal of edge computing is to reduce latency requirements while processing data and saving network costs.

The main purpose of edge computing is to reduce latency, improve real-time processing, and enhance efficiency for applications that require immediate data analysis, rapid response time.

Definition: Edge computing refers to the practice of processing and analyzing data closer to the data source or the "edge" of the network, rather than sending all the data to a centralized cloud data center. This approach aims to minimize the latency caused by sending data to distant cloud servers for processing. Instead, edge devices or edge servers process the data locally or in nearby locations, enabling faster decision-making and more efficient resource utilization.

How Does Edge Computing Work:

In a traditional setting, data is produced on a user's computer or any other client application. It is then moved to the server through channels like the internet, intranet, LAN, etc., where the data is stored and worked upon. This remains a classic and proven approach to client-server computing.

However, the exponential growth in the volume of data produced and the number of devices connected to the internet has made it difficult for traditional data center infrastructures to accommodate them.

The concept of edge computing is simple - instead of getting the data close to the data center, the data center is brought close to the data. The storage and computing resources from the data center are deployed as close as possible (ideally in the same location) to where the data is generated.

Benefits of Edge Computing:

Eliminates Latency: By processing data closer to its source, edge computing reduces the time it takes for data to travel to a central cloud server and back. This results in lower latency, making it ideal for applications that require real-time responses, such as industrial automation, autonomous vehicles, and augmented reality.

Faster Data Processing: Edge devices can perform immediate data analysis and decision-making locally, without needing to transmit data to a remote server.

Bandwidth Efficiency: Instead of sending large volumes of raw data to a cloud data center, edge computing allows for pre-processing and filtering of data at the edge. Only relevant information is sent to the cloud, reducing the need for high bandwidth

Enhanced Privacy and Security: Edge computing allows sensitive data to be processed locally, reducing the need to transmit private information to external servers.

Offline Capabilities: Edge devices can continue to operate and process data even when disconnected from the central cloud.

Real-time Decision-Making: For applications that require immediate actions based on data analysis, edge computing enables real-time decision-making without waiting for data to travel to a remote data center.

Reduced Network Congestion/traffic: Processing data at the edge helps network congestion by minimizing the amount of data that needs to traverse the network. This is especially important in scenarios with limited network capacity

Types of Edge Computing:

Internet of Things (IoT) Edge: Latency expectations at this Edge is usually less than 1 millisecond.

This is like having your gadgets communicate directly with each other. IoT edge computing involves processing data from IoT devices (like smart sensors, cameras) right where they're located, without sending everything to a central location. This reduces the need to send all the data to a central location, making processing faster and more efficient. This speeds up response times, conserves bandwidth, and is perfect for scenarios where quick actions matter, like smart homes, industrial automation, and connected vehicles.

On-Premise Edge: Latency expectations at this Edge is usually less than 5 milliseconds.

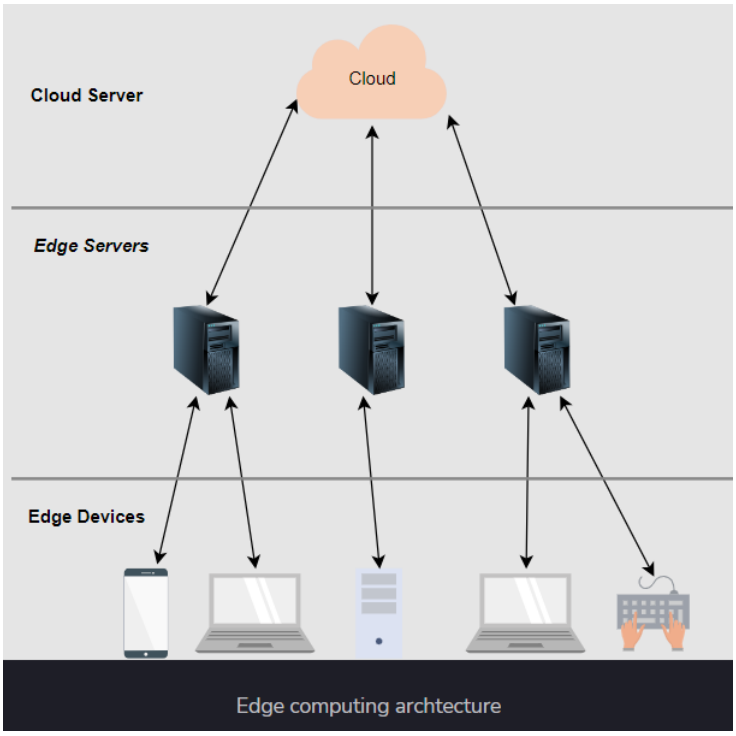
Instead of sending your data somewhere far, you process it using local devices you own, like your own personal servers. This keeps your data close, reduces delays, and gives you more control over your computing resources.

Access Edge: The latency expectation at this Edge is usually less than 10–40 milliseconds.

Access Edge in edge computing is like a neighborhood hub where people come together to connect. It's about processing data at the entry points of a network – where devices and users access it. This helps optimize data processing right where

it's needed, reducing the load on central systems and improving response times for better overall performance. It's like having a mini data processing station at the entrance to your digital neighborhood.
Network Edge: The latency expectation at this Edge is usually less than 10–40 milliseconds as well.
Network Edge in edge computing is like having a mini data processing station near where you are. It involves processing data close to the location where devices or users connect to the network. This reduces the time data takes to travel and speeds up responses. Think of it as getting faster service at a neighborhood coffee shop rather than going to a distant city café. Network edge computing improves performance by handling data closer to where it's generated or needed, making applications more responsive and efficient.

Edge Computing Architecture:



Edge computing Components:
Edge Devices:
These are devices like IoT sensors, cameras, and local servers located close to where data is generated. They collect and preprocess data before sending it for further analysis.
Edge Gateway:
This acts as a bridge between edge devices and the cloud. It aggregates data from multiple devices, applies more complex analytics, and communicates with both local and remote systems.
Edge Nodes or Servers:
These are computing nodes or servers located closer to edge devices. They perform initial data processing, filtering, and analysis before sending relevant information to higher-level systems.
Cloud: A centralized server that does more complex computational work. It also helps in hosting the applications for edge users. It could be private or public, and it stores models like the neuron network that helps applications do their tasks.

Edge Computing Use-Cases:
Industrial IoT (IIoT): Monitoring and controlling machinery in factories for efficient production.
Smart Cities: Optimizing traffic, waste management, and public services in urban areas.
Autonomous Vehicles: Enabling quick decision-making in self-driving cars for safety.
Retail Analytics: Analyzing customer behavior in stores for better shopping experiences.
Healthcare Monitoring: Real-time tracking of health data in wearable devices.
Remote Monitoring: Data collection and analysis in remote locations for environmental monitoring.
Energy Management: Efficient management of power generation and distribution.
Video Surveillance: Local analysis of video feeds for enhanced security.
Agriculture: Monitoring soil and crop conditions for better yields.
Emergency Response: Providing critical data to first responders during emergencies.
Retail Point-of-Sale (POS): Quick and smooth transactions at checkout counters.

CHAPTER 5:

Fog Computing: Fog computing is computing that uses devices at the edge of a network (like IoT devices, sensors, or local servers) to do a big part of the computing, storage, and communication work right where they are. Instead of sending everything far away to the central internet, these devices handle a lot of tasks nearby and only send what's necessary over the main internet connection.

Cloud Computing vs Fog Computing:

Specialty	Cloud Computing	fog computing
Delay	Cloud computing has higher latency than fog computing	Fog computing has low latency
Capacity	Cloud computing does not provide any reduction in data while sending or converting data.	Fog computing reduces the amount of data sent to cloud computing.
Responsiveness	The response time of the system is low.	The response time of the system is high.
Security	Cloud computing has less Security compared to Fog Computing	Fog computing has high Security.
Speed	Access speed is high depending on the VM connectivity.	High even more compared to Cloud Computing.
Data Integration	Multiple data sources can be integrated.	Multiple Data sources and devices can be integrated.
Mobility	In cloud computing, mobility is Limited.	Mobility is supported in fog computing.
Location Awareness	Partially Supported in Cloud computing.	Supported in fog computing.
Number of Server Nodes	Cloud computing has Few numbers server nodes.	Fog computing has a Large number of server nodes.
Geographical Distribution	It is centralized.	It is decentralized and distributed.
Location of service	Services provided within the Internet.	Services are provided at the edge of the local network.
Working environment	Specific data center building with air conditioning systems	Outdoor (streets, base stations, etc.) or indoor (houses, cafes, etc.)
Communication mode	IP network	Wireless communication: WLAN, WiFi, 3G, 4G, ZigBee, etc. or wired communication (part of the IP networks)
Dependence on the quality of core network	Requires strong network core.	It can also work in a Weak network core.

Edge Computing vs Fog Computing:

EDGE COMPUTING	FOG COMPUTING
Less scalable than fog computing.	Highly scalable when compared to edge computing.
Billions of nodes are present.	Millions of nodes are present.
Nodes are installed far away from the cloud.	Nodes in this computing are installed closer to the cloud(remote database where data is stored).
Edge computing is a subdivision of fog computing.	Fog computing is a subdivision of cloud computing.
The bandwidth requirement is very low. Because data comes from the edge nodes themselves.	The bandwidth requirement is high. Data originating from edge nodes is transferred to the cloud.
Operational cost is higher.	Operational cost is comparatively lower.
High privacy. Attacks on data are very low.	The probability of data attacks is higher.
Edge devices are the inclusion of the IoT devices or client's network.	Fog is an extended layer of cloud.
The power consumption of nodes is low.	The power consumption of nodes filter important information from the massive amount of data collected from the device and saves it in the filter high.
Edge computing helps devices to get faster results by processing the data simultaneously received from the devices.	Fog computing helps in filtering important information from the massive amount of data collected from the device and saves it in the cloud by sending the filtered data.

When to use fog computing?

- It is used when only selected data is required to send to the cloud. This selected data is chosen for long-term storage and is less frequently accessed by the host.
- It is used when the data should be analyzed within a fraction of seconds i.e Latency should be low.
- It is used whenever a large number of services need to be provided over a large area at different geographical locations.
- Devices that are subjected to rigorous computations and processings must use fog computing.

Applications of fog computing:

- It can be used to monitor and analyze the patients' condition. In case of emergency, doctors can be alerted.
- It can be used for real-time rail monitoring as for high-speed trains we want as little latency as possible.
- It can be used for gas and oils pipeline optimization. It generates a huge amount of data and it is inefficient to store all data into the cloud for analysis.

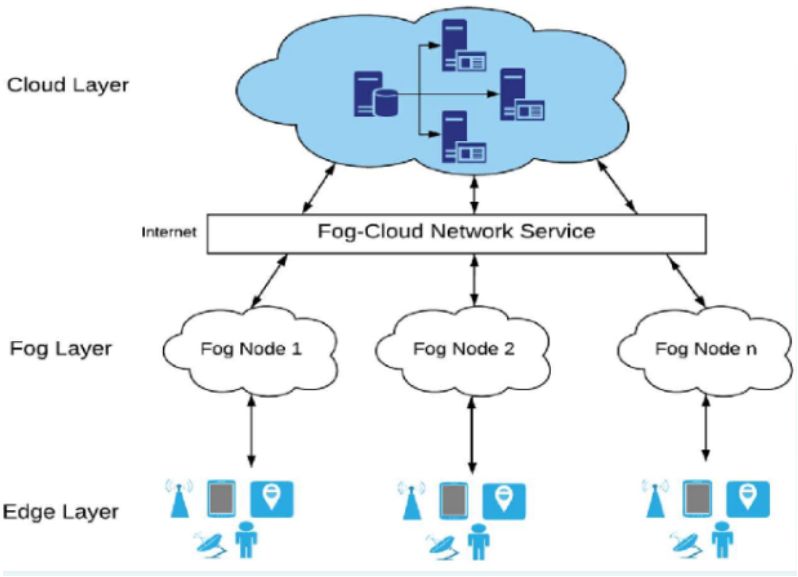
Advantages of Fog Computing

- This approach reduces the amount of data that needs to be sent to the cloud.
- Since the distance to be traveled by the data is reduced, it results in saving network bandwidth.
- Reduces the response time of the system.
- It improves the overall security of the system as the data resides close to the host.
- It provides better privacy as industries can perform analysis on their data locally.

Disadvantages of Fog Computing

- Congestion may occur between the host and the fog node due to increased traffic (heavy data flow).
- Power consumption increases when another layer is placed between the host and the cloud.
- Scheduling tasks between host and fog nodes along with fog nodes and the cloud is difficult.
- Data management becomes tedious as along with the data stored and computed, the transmission of data involves encryption-decryption too which in turn release data.

Architecture of Fog Computing:



- Fog computing is a type of distributed computing that connects a cloud to a number of "peripheral" devices. (The term "fog" refers to the edge or perimeter of a cloud.)
- Rather than sending all of this data to cloud-based servers to be processed, many of these devices will create large amounts of raw data (for example, via sensors).
- The goal of fog computing is to conduct as much processing as possible using computing units that are co-located with data-generating devices so that processed data rather than raw data is sent and bandwidth needs are decreased.

Fog Computing Components

Physical & virtual nodes (end devices): End devices serve as the points of contact to the real world, be it application servers, edge routers, end devices such as mobile phones and smartwatches, or sensors.
Fog nodes: Fog nodes are independent devices that pick up the generated information.
Monitoring services: Monitoring services usually include application programming interfaces (APIs) that keep track of the system's performance and resource availability.
Data processors: Data processors are programs that run on fog nodes. They filter, trim, and sometimes even reconstruct faulty data that flows from end devices.
Resource manager: The resource manager allocates and deallocates resources to various nodes and schedules data transfer between nodes and the cloud.
Security tools: Since fog components directly interact with raw data sources, security must be built into the system even at the ground level.
Applications: Applications provide actual services to end-users.

Major Issues with Fog Computing

Authentication and Trust issues: Authentication is one of the most concerning issues of fog computing since these services are offered at a large scale.

Privacy: Privacy concern is always there when there are many networks involved. Since fog computing is based on wireless technology, there is a huge concern regarding network privacy.
Security: Fog computing security issues arise as there are many devices connected to fog nodes and at different gateways.
Fog Servers: The right placement of fog servers should be there so that it can deliver its maximum service.
Energy consumption: Energy consumption is very high in fog computing as the number of fog nodes present in the fog environment are high and require energy to work.

CHARACTERISTICS OF THE FOG COMPUTING
Heterogeneity: Heterogeneity in fog computing means that there's a mix of different devices, systems working together.
Real-Time Processing: Fog computing supports immediate data analysis and decision-making, crucial for time-sensitive applications.
Low Latency: Processing data locally or within the network minimizes latency, enabling quicker responses for real-time applications.
Local Decision-Making: Devices at the edge can make decisions locally, reducing the need for constant communication with central servers.
Bandwidth Efficiency: By processing data locally, fog computing reduces the demand on network bandwidth and lowers data transmission costs.
Enhanced Privacy: Sensitive data can be processed locally, reducing the risk of data exposure during transmission.
Support for mobility: It is essential for many Fog applications to communicate directly with mobile devices, and therefore support mobility techniques.
Distributed Architecture: Fog computing involves a distributed network of edge devices, local servers, and gateways that work together to process data.

CHAPTER 6:

NameSpace: In Linux, namespaces are a feature of the kernel that provides process and resource isolation
They allow multiple instances of certain system resources to coexist independently within their own isolated namespaces.
Linux namespaces include several types, each dedicated to isolating specific resources:
- *PID Namespace (pid)*: This namespace isolates processes. Each PID namespace has its own set of process IDs (PIDs), so processes in different PID namespaces can have the same PID without conflicts. This is essential for containerization, as it allows containers to run their own processes without interfering with the host or other containers.
- *Network Namespace (net)*: Network namespaces isolate network resources, such as network interfaces, IP addresses, routing tables, and firewall rules. This enables network isolation between containers and the host.
- *Mount Namespace (mnt)*: Mount namespaces isolate the file system mount points. This enables file system isolation and allows containers to have their own root file system.
- *UTS Namespace (uts)*: UTS namespaces isolate the hostname and NIS domain name. Each UTS namespace allows processes to have their own hostname, which is useful for containerization to provide a distinct identity for containers.
- *IPC Namespace (ipc)*: IPC namespaces isolate inter-process communication mechanisms, such as System V IPC objects (semaphores, shared memory, and message queues). Each IPC namespace allows processes to use IPC objects independently from processes in other namespaces.
- *User Namespace (user)*: User namespaces provide a way to map user and group IDs from the container's perspective to those on the host.User namespaces enhance container security by reducing the privileges within the container.

CGROUPS: Cgroups are a Linux kernel feature that enables control and allocation of system resources among processes and groups of processes.
Control Groups, commonly referred to as cgroups, are a crucial feature in advanced computing, particularly in the context of resource management, resource isolation, and performance tuning
Here are some key aspects of cgroups in advanced computing:
- Resource Management: Cgroups allow administrators to allocate and control resources such as CPU time, memory, I/O bandwidth, and network bandwidth to groups of processes or individual processes.
- Isolation: Cgroups provide process isolation by grouping processes together and limiting their resource usage.
- Prioritization: Cgroups enable administrators to set priorities and allocate resources based on the importance of processes
- Resource Accounting: Cgroups also offer resource usage tracking and accounting.

LAYERED FILESYSTEMS: optimal way to make a copy of root filesystem for each container

WHAT ARE CONTAINERS ?: Containers are lightweight packages of your application code together with dependencies such as specific versions of programming language runtimes and libraries required to run your software services.
- Containers in advanced computing are lightweight, portable, and self-sufficient environments for running applications and services. They package all the necessary dependencies, libraries, and configuration files into a single unit, making it easy to deploy and run software consistently across different environments, such as development, testing, and production.
- Containers make it easy to share CPU, memory, storage, and network resources at the operating systems level

PROS OF USING CONTAINERS:
- Isolation: Containers provide process and file system isolation, ensuring that applications and their dependencies run independently of each other and the host system.
- Portability: Containers package applications and all their dependencies, including libraries, into a single, consistent unit called a container image. These images can be easily moved between different environments, such as development, testing, and production.
- Consistency: Containers enable consistent application behavior across diverse environments,
- Scalability: Containers are highly scalable, both vertically (by allocating more resources to a container) and horizontally (by adding more instances of containers).
- Easy to maintain
- Effective resource usage

CONS OF USING CONTAINERS:
- Lacking Security measures: Containers provide lightweight isolation from the host OS and containers within the same system. This leads to a weaker security boundary compared to virtual machines
- Runs only one OS :- This can be a benefit if you only use one OS, but if you need to be able to use it across different OS's this is a negative. You can run an earlier version of the same OS using lightweight virtual machines.
- Complex Networking: Networking in containerized environments can be complex, especially when dealing with multi-container applications or services that need to communicate across different containers or hosts
- Data Management: Handling data, especially in stateful applications, can be complex.

POPULAR CONTAINER PROVIDERS
RedHat Openshift
Google Kubernete Engine
Docker
Linux Containers (LXC, LXD, CGManager)
Mesos
Windows Server Containers

IMPLEMENTING CONTAINER
1.run input commands with arguments :-To run a command with arguments in a container, you can use the 'unshare' command to create a new namespace, and then execute the desired command within that namespace.
Example :-
```
# Create a new PID and mount namespace
unshare --fork --pid --mount-proc /bin/bash -c "your-command arg1 arg2"
```
2. add hostname limitations :- To limit the hostname within the container, you can set the hostname using the hostname command within the container namespace:
3. add process ID limitations :- To limit the number of processes within the container, you can use cgroups to control process counts.
```
# Set the maximum number of processes
cgset -r pids.max=100 /mycontainer
```
4. add mount table/filesystem limitations :- To limit the mount table and filesystem access within the container, you can create a separate mount namespace and restrict filesystem access by chrooting into a specific directory.

Virtual Machine vs Containers:

Virtual Machines(VM)	Containers
VM is piece of software that allows you to install other software inside of it so you basically control it virtually as opposed to installing the software directly on the computer.	While a container is a software that allows different functionalities of an application independently.
Applications running on VM system can run different OS.	While applications running in a container environment share a single OS.
VM virtualizes the computer system.	While containers virtualize the operating system only.
VM size is very large.	While the size of container is very light; i.e. a few megabytes.
VM takes minutes to run, due to large size.	While containers take a few seconds to run.
VM uses a lot of system memory.	While containers require very less memory.
VM is more secure.	While containers are less secure.
VM's are useful when we require all of OS resources to run various applications.	While containers are useful when we are required to maximise the running applications using minimal servers.

Feature	Virtual Machines (VMs)	Containers
Isolation	VMs offer strong isolation since they run separate guest OS instances.	Containers provide lightweight isolation, sharing the host OS kernel.
Resource Overhead	VMs have higher resource overhead due to emulating hardware for each instance.	Containers have lower resource overhead since they share the host OS kernel.
Startup Time	VMs generally have longer startup times as they involve booting a complete OS.	Containers have very fast startup times since they use the existing OS.
Resource Utilization	VMs are less efficient at resource utilization because of the hypervisor layer.	Containers are highly efficient, maximizing resource utilization.
Scalability	VMs can be less flexible to scale quickly due to their larger size.	Containers are highly scalable and can be rapidly provisioned.
Isolation Level	VMs provide strong isolation, making them suitable for running different OSs.	Containers provide moderate isolation, suitable for applications sharing the same OS.
Portability	VMs can be less portable due to differences in guest OS and hypervisor requirements.	Containers are highly portable because they package applications and dependencies.
Security	VMs offer strong security isolation, which is beneficial for multi-tenant environments.	Containers offer less isolation but can still provide security through best practices.
Resource Sharing	VMs can't easily share resources like libraries or memory between instances.	Containers can share resources, making them more memory-efficient.
Orchestration Tools	VMs are typically managed using hypervisor-specific tools (e.g., VMware vCenter, Hyper-V Manager).	Containers are managed using container orchestration platforms (e.g., Kubernetes, Docker Swarm).
Use Cases	VMs are suitable for running applications with varying OS requirements, legacy systems, or multi-tenant environments.	Containers are ideal for microservices, cloud-native applications, and efficient resource utilization.

What is Docker?:
- Docker is an open-source platform for developing, packaging, and deploying applications inside lightweight, portable containers.
- Docker containers provide a consistent environment for running applications and their dependencies, making it easier to develop, test, and deploy software across different computing environments, from development laptops to production servers.
- Docker is a centralized platform for packaging, deploying, and running applications. Before Docker, many users faced the problem that a particular code is running in the developer's system but not in the user's system. So, the main reason to develop docker is to help developers to develop applications easily, ship them into containers, and can be deployed anywhere.
- It includes components such as Docker client, Docker server, Docker machine, Docker hub, Docker composes, etc.
- Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.
- Docker uses a container on the host's operating system to run applications. It allows applications to use the same Linux kernel as a system on the host computer, rather than creating a whole virtual operating system.

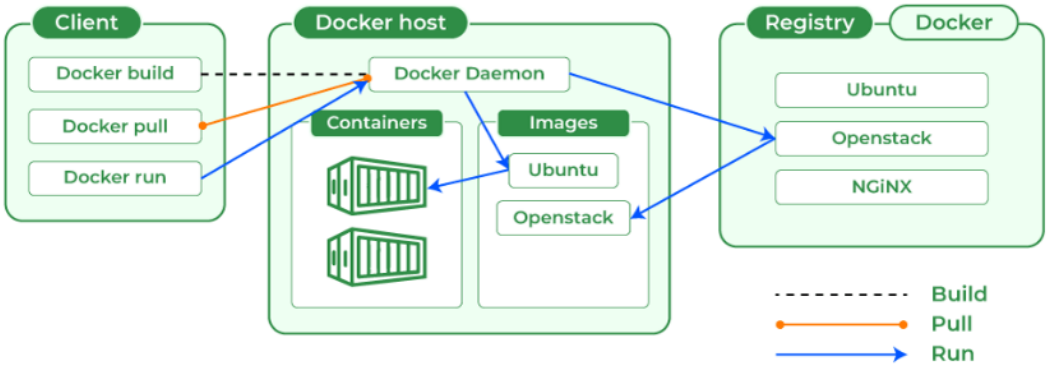
WHY DOCKER ?
- Docker is designed to benefit both the Developer and System Administrator. There are the following reasons to use Docker
- Docker allows us to easily install and run software without worrying about setup or dependencies.
- Developers use Docker to eliminate machine problems, i.e. "but code is worked on my laptop." when working on code together with co-workers.
- Operators use Docker to run and manage apps in isolated containers for better compute density.
- Enterprises use Docker to securely built agile software delivery pipelines to ship new application features faster and more securely.
- Apart from it, Dockers features like Portability, Efficiency, Containerization, Rapid Development that’s why Docker is used.
- Since docker is not only used for the deployment, but it is also a great platform for development, that's why we can efficiently increase our customer's satisfaction.

Docker Features:
- Easy and Faster Configuration: This is a key feature of docker that helps us to configure the system easily and faster.
- Increase productivity: By easing technical configuration and rapid deployment of application. No doubt it has increase productivity
- Application Isolation: It provides containers that are used to run applications in isolation environment. Each container is independent of another and allows us to execute any kind of application.
- Swarm: It is a clustering and scheduling tool for Docker containers. Swarm uses the Docker API as its front end, which helps us to use various tools to control it. It also helps us to control a cluster of Docker hosts as a single virtual host. It's a self-organizing group of engines that is used to enable pluggable backends.
- Routing Mesh: It routes the incoming requests for published ports on available nodes to an active container. This feature enables the connection even if there is no task running on the node.
- Cross-Platform: Docker containers are designed to be platform-agnostic, meaning that containers built on one system can run on another without modification, as long as both systems support Docker.
- Security: Docker containers offer security isolation, although they are not as strong as virtual machines

ADVANTAGES OF DOCKER:
It runs the container in seconds instead of minutes.
It uses less memory.
It provides lightweight virtualization.
It does not a require full operating system to run applications.
It uses application dependencies to reduce the risk.
Docker allows you to use a remote repository to share your container with others.
It provides continuous deployment and testing environment.

DISADVANTAGES OF DOCKER:
- It increases complexity due to an additional layer.
- In Docker, it is difficult to manage large amount of containers.
- Some features such as container self -registration, containers self-inspects, copying files form host to the container, and more are missing in the Docker.
- Docker is not a good solution for applications that require rich graphical interface.
- Docker provides cross-platform compatibility means if an application is designed to run in a Docker container on Windows, then it can't run on Linux or vice versa.

WHAT IS DOCKER DAEMON?: Docker daemon runs on the host operating system and manages Docker Containers on a host system. It is responsible for running containers to manage docker services. Docker daemon communicates with other daemons. It offers various Docker objects such as images, containers, networking, and storage.
DOCKER ARCHITECTURE:
Docker follows Client-Server architecture, which includes the three main components that are Docker Client, Docker Host, and Docker Registry.



1. Docker Client: Docker client uses commands and REST APIs to communicate with the Docker Daemon (Server). When a client runs any docker command on the docker client terminal, the client terminal sends these docker commands to the Docker daemon. Docker daemon receives these commands from the docker client in the form of command and REST API's request.
- Docker Client has an ability to communicate with more than one docker daemon.
- Docker Client uses Command Line Interface (CLI) to run the following commands -
docker build
docker pull
docker run
2. Docker Host: Docker Host is used to provide an environment to execute and run applications. It contains the docker daemon, images, containers, networks, and storage.
3. Docker Registry: Docker Registry manages and stores the Docker images.
There are two types of registries in the Docker -
Public Registry - Public Registry is also called Docker hub.
Private Registry - It is used to share images within the enterprise.

DOCKER OBJECTS
There are the following Docker Objects -
Docker Images: Docker images are the read-only binary templates used to create Docker Containers.It uses a private container registry to share container images within the enterprise and also uses a public container registry to share container images within the whole world. Metadata is also used by docket images to describe the container's abilities.
Docker Containers: Containers are the structural units of Docker, which is used to hold the entire package that is needed to run the application. The advantage of containers is that it requires very less resources.
In other words, we can say that the image is a template, and the container is a copy of that template.

Docker Networking:
Using Docker Networking, an isolated package can be communicated. Docker contains the following network drivers -
Bridge - Bridge is a default network driver for the container. It is used when multiple docker communicates with the same docker host.
Host - It is used when we don't need network isolation between the container and the host.
None - It disables all the networking.
Overlay - Overlay offers Swarm services to communicate with each other. It enables containers to run on the different docker hosts.
Macvlan - Macvlan is used when we want to assign MAC addresses to the containers.

Docker Storage:
Docker Storage is used to store data on the container. Docker offers the following options for the Storage -
Data Volume - Data Volume provides the ability to create persistence storage. It also allows us to name volumes, list volumes, and containers associated with the volumes.
Directory Mounts - It is one of the best options for docker storage. It mounts a host's directory into a container.
Storage Plugins - It provides an ability to connect to external storage platforms.

DOCKER COMPOSE:
- It is a tool which is used to create and start a Docker application by using a single command. We can use it to file to configure our application's services.
- When using Docker extensively, the management of several different containers quickly becomes unmanageable. Docker Compose is a tool that helps us overcome this problem and easily handle multiple containers at once.
- It provides the following commands for managing the whole lifecycle of our application.
Start, stop and rebuild services
View the status of running services
Stream the log output of running services
Run a one-off command on a service

WHAT IS CONTAINER ORCHESTRATION: Container orchestration is the process of automating the deployment, scaling, management, and maintenance of containerized applications.
- It involves managing the lifecycle of containers, including their creation, placement on host machines, coordination of communication between containers, scaling based on demand, and ensuring high availability and fault tolerance.
- It automates tasks such as starting, stopping, scaling, and updating containers, ensuring they work together smoothly
- Popular container orchestration platforms include Kubernetes and Docker Swarm.
Key tasks in container orchestration include:
1. Scheduling: Deciding where to run containers across multiple machines (nodes) to balance the workload and ensure high availability.
2. Scaling: Automatically adjusting the number of containers to handle changes in traffic or demand.
3. Load Balancing: Distributing incoming requests or work evenly across containers to prevent overload on any one part of the application.
4. Health Checking: Monitoring the health of containers and replacing or repairing them if they fail.

WHY WE NEED CONTAINER ORCHESTRATION ?: We need container orchestration because it helps us manage and run our containers in a more organized and efficient way.
Here's why container orchestration is essential:
Scaling Up and Down:container orchestration allows you to scale your containers up (add more) or down (reduce) based on traffic or demand
High Availability: Container orchestration detects failures and automatically replaces or repairs containers to keep your application running smoothly.
Load Balancing:container orchestration balances incoming requests or work across containers, preventing any one part of your application from getting overwhelmed.

WHAT IS KUBERNETS CONTAINER ORCHESTRATION ?Kubernetes is a popular open source platform for container orchestration. It enables developers to easily build containerized applications and services, as well as scale, schedule and monitor those containers.
Kubernetes includes various features such as:
- runs everywhere
- automated rollouts and rollback: Using the rollouts, Kubernetes distributes the changes and updates to an application or its configuration. If any problem occurs in the system, then this technique rollbacks those changes for you immediately.
- storage orchestration
- Batch execution, secret and configuration management
- horizontal scaling
- Pod: It is a deployment unit in Kubernetes with a single Internet protocol address.
- Persistent Storage: Kubernetes provides an essential feature called 'persistent storage' for storing the data, which cannot be lost after the pod is killed or rescheduled.
- Service Discovery and load balancing: Kubernetes assigns the IP addresses and a Name of DNS for a set of containers, and also balances the load across them.
- Self-Healing: This feature plays an important role in the concept of Kubernetes. Those containers which are failed during the execution process, Kubernetes restarts them automatically.

The advantage of using Kubernetes is that it provides the best solution for scaling up the containers.

CHAPTER :7
Notes Link: https://drive.google.com/file/d/14MBatB--0RjcdN7vDQV31qGULqTZELK5/view?usp=sharing

UNIT TEST PAPER:

Unit test 1: https://drive.google.com/file/d/1qX6cw0FxR_LgDqIliquEqGJPMAbCR7IC/view?usp=sharing
Unit test 2: https://drive.google.com/file/d/1JWfG0-tWz1wiV2VTWVFJbEObxLurwCBU/view?usp=sharing
(NOTE: in MCQ 3rd MCQs ANSWER IS D NOT C)

QUESTIONBANK:

https://drive.google.com/file/d/1IGE_xeMJ4ZPWuuIxvxr3FMVNfnJ7_94H/view?usp=sharing

REMAINING QUESTIONS FROM QUESTION BANK:

What is Virtual Machine? Explain Virtual Machine types.
https://www.geeksforgeeks.org/types-of-virtual-machines/

What is Virtualization? What is the role of virtualization in cloud computing? Which form of virtualization is used in the cloud for effective resource utilization?
https://www.javatpoint.com/virtualization-in-cloud-computing
Virtualization is a technology that enables the creation of virtual instances of computing resources, such as servers, storage, and networks. It abstracts the physical hardware, allowing multiple virtual instances to run on a single physical machine. This can enhance resource utilization, improve flexibility, and simplify management.

What are service elasticity and service orchestration in cloud computing? What are the issues of Software as a Service (SaaS)?
Service Elasticity: Service elasticity in cloud computing refers to the ability of a cloud service or application to dynamically scale its resources up or down based on demand. This scalability allows for the efficient use of resources, ensuring that the service can handle varying workloads and adapt to changing requirements.
Key features of service elasticity include:
Automatic Scaling: Cloud services can automatically adjust the number of resources (e.g., virtual machines, storage, or network capacity) based on factors such as traffic volume, computational load, or other performance metrics.

On-Demand Provisioning: Resources are provisioned or de-provisioned in real-time as needed, allowing the service to handle peak loads without over-provisioning during periods of lower demand.
Cost Efficiency: Elasticity contributes to cost efficiency by enabling organizations to pay only for the resources they actually use.

Service Orchestration:

Service orchestration involves the coordination and management of multiple cloud services or components to achieve a specific business process or workflow.
Key aspects of service orchestration include:
Workflow Automation: Orchestration involves automating the execution of predefined workflows or processes that span multiple services. This can include provisioning, configuration, and deployment of services.
Integration of Services: Orchestration allows different cloud services, APIs, and components to work together cohesively, creating a unified solution. This is particularly important in complex applications where multiple services need to interact.
Dynamic Adaptation: Orchestration frameworks often include mechanisms for dynamic adaptation, enabling adjustments to the workflow in response to changing conditions or requirements. This contributes to agility and flexibility in managing complex business processes.
End-to-End Management: Service orchestration provides end-to-end management of workflows, ensuring that tasks are executed in a coordinated manner, dependencies are managed, and errors or exceptions are handled appropriately.

issues of Software as a Service (SaaS):

Limited Control: Organizations using SaaS have less control over the underlying infrastructure, application updates, and security protocols. This lack of control can be a concern for businesses with specific regulatory or compliance requirements.
Vendor Lock-In: Switching from one SaaS provider to another can be challenging due to data formats and dependencies.
Limited Customization: SaaS applications are often standardized, and users may have limited ability to customize features to meet specific needs.
Data Security Concerns: Storing data in the cloud can raise security concerns. Users may worry about the safety of their sensitive information.
Offline Access Limitations: Some SaaS applications require an internet connection, making offline access challenging.
Service Outages: SaaS services are dependent on the provider's infrastructure, and occasional outages can occur.

Which are the deployment models of cloud? How do private cloud works? CHARUSAT University wants to manage and Analyze Engineering student's record on a cloud. Which Deployment is suitable for this scenario? Why?

Deployment models of Cloud: Public Cloud, Private Cloud, Hybrid Cloud & Community Cloud
In the scenario of CHARUSAT University wanting to manage and analyze engineering student records on a cloud, a Hybrid Cloud Deployment would be suitable.
A hybrid cloud is like having two different types of clouds that work together. One part is like a private cloud that is just for CHARUSAT University (it's like their own special cloud), and the other part is like a public cloud that many people can use.
Why Hybrid Cloud for CHARUSAT University:
Security for Student Records: Student records often have sensitive information. With a hybrid cloud, CHARUSAT can keep the most sensitive data (like personal details) in their private cloud, giving them more control and security.
Flexible Analysis Tools in the Public Cloud: To analyze student data, CHARUSAT might want to use powerful tools available in the public cloud. With a hybrid setup, they can do this without putting all the student records on the public cloud.
Handling Peak Loads: During busy times, like exams or admissions, CHARUSAT might need extra computing power. With a hybrid cloud, they can use the public cloud to handle these peak loads without buying a lot of extra equipment.
Cost Efficiency: Running a private cloud can be cost-effective for regular, everyday tasks. When they need extra resources for specific tasks, they can use the public cloud, paying only for what they use.
Gradual Transition to Cloud: If CHARUSAT is just starting to use the cloud, a hybrid approach allows them to move at their own pace. They can gradually move some things to the cloud without making big changes all at once.

So, in simple terms, a hybrid cloud for CHARUSAT University provides a mix of security, flexibility, and cost-effectiveness, allowing them to manage and analyze engineering student records in a way that suits their needs and priorities.

Which are the service models of cloud? Which service model is suitable for storing the large amount of data and performing heavy computation tasks of a Scientific research lab? Justify your answer.

The three main service models in cloud computing are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

For a Scientific Research Lab dealing with storing a large amount of data and performing heavy computation tasks, the most suitable service model is Infrastructure as a Service (IaaS).

Reasons for Choosing IaaS:

Customization and Control:IaaS provides the lab with the highest level of control over the infrastructure. Researchers can configure virtual machines and storage according to their specific requirements, optimizing the environment for their scientific computations.

Scalability:Scientific research often involves large datasets and complex computations. IaaS allows the lab to scale up resources when needed and scale down when the demand decreases. This flexibility is crucial for handling varying workloads associated with scientific research.

Storage Capacity:IaaS offers scalable storage solutions, allowing the lab to store a large amount of data.

Heavy Computation Tasks:IaaS is well-suited for heavy computation tasks as it provides access to virtual machines with significant computational power. Researchers can deploy high-performance computing instances for running complex simulations, data analyses, or modeling tasks.

Integration with Existing Tools:Research labs often have specific tools and software requirements. With IaaS, the lab can use virtual machines with different operating systems, software stacks, and configurations, enabling seamless integration with existing tools used in scientific research.

Cost Efficiency:IaaS allows the lab to pay for resources on a pay-as-you-go basis.

Explain Taxonomy of Virtualization in brief.

https://youtu.be/7yXlLgZkiJM

https://youtu.be/2rwUGh6GDjM

Explain in brief about Cluster middleware and Single System Image?

Cluster Middleware: Cluster middleware refers to software that facilitates communication and coordination among nodes (computers) in a cluster, which is a group of interconnected computers that work together to perform tasks. In cloud computing, clusters are often used to distribute workloads and enhance the overall performance and reliability of applications.

Key characteristics of cluster middleware include:

Communication: Cluster middleware enables communication between nodes in the cluster. This communication is crucial for coordinating tasks, sharing data.

Load Balancing: Middleware helps distribute the workload across the cluster nodes to ensure that resources are utilized efficiently.

Fault Tolerance: Cluster middleware often includes mechanisms for detecting and handling faults. If a node in the cluster fails, the middleware can redistribute tasks to healthy nodes to maintain system reliability.

Scalability: Cluster middleware supports the scalability of applications by allowing the addition of new nodes to the cluster as needed.

Single System Image (SSI):

A Single System Image (SSI) in cloud computing refers to the perception of a collection of distributed and possibly heterogeneous resources as a single, unified system. From the user's perspective, the SSI approach makes the cluster or distributed system appear as a single, cohesive entity.

Unified Resource Management: SSI systems provide a unified view and management of resources such as memory, storage, and processing power. Users can access and manage these resources seamlessly.

Transparency: Users interact with the system as if it were a single, powerful computer, regardless of the actual physical distribution of resources. This transparency simplifies the user experience.

Fault Tolerance: SSI systems often include fault-tolerance mechanisms. If a node fails, the system can automatically redistribute tasks to maintain continuity of service.
Process Migration: SSI systems may support the migration of processes across nodes in the cluster. This allows for dynamic load balancing and efficient utilization of resources.

Explain in detail about Resource Pooling Architecture with diagram?
https://www.geeksforgeeks.org/resource-pooling-architecture-in-cloud-computing/

Explain in detail about Dynamic Scalability Architecture with diagram?
https://www.informit.com/articles/article.aspx?p=2093407&seqNum=3

Explain in detail about Elastic Resource Capacity Architecture with diagram?
https://www.informit.com/articles/article.aspx?p=2093407&seqNum=4

Explain Layered Grid Architecture & Service Modeling.

Explain different Grid data access models.
https://www.slideshare.net/gomathynayagam/data-intensive-grid-service-model *(From Page no. 5 in Slide)*

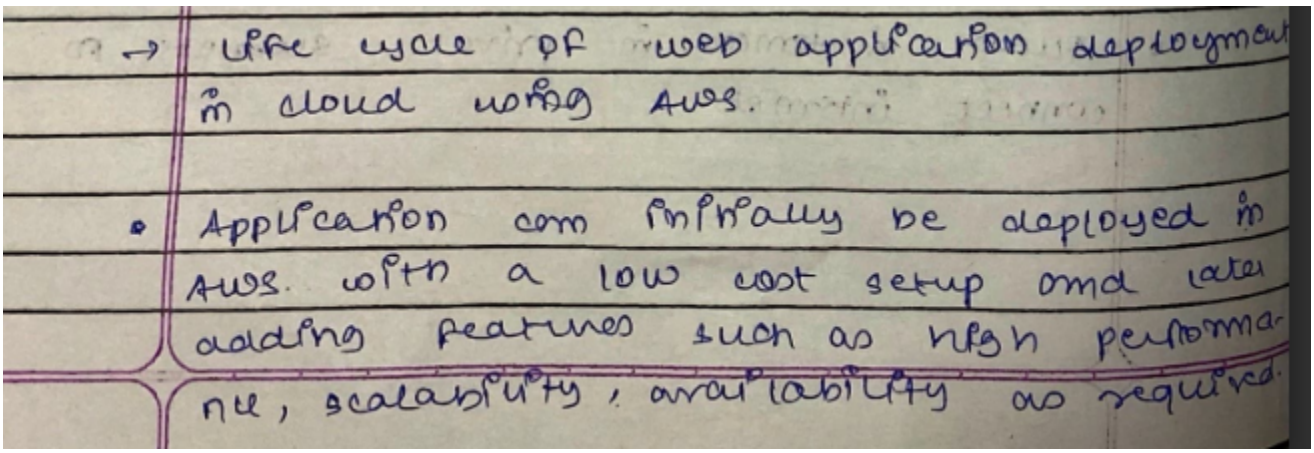
Describe in detail about Load balanced virtual server instance architecture.

Draw and explain dynamic load balancing architecture in detail?
https://www.geeksforgeeks.org/static-vs-dynamic-load-balancing/

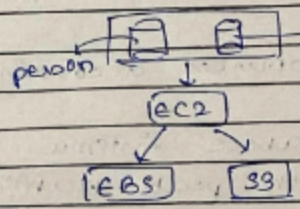
What are the roles of Cloud Bursting Architecture? Explain in detail.
https://www.informit.com/articles/article.aspx?p=2093407&seqNum=6

Explain in detail about Data center Technology used as Cloud enabled Technology?

Explain lifecycle of web application deployment in the Cloud using Amazon Web Services (AWS) with auto scaling and high availability.



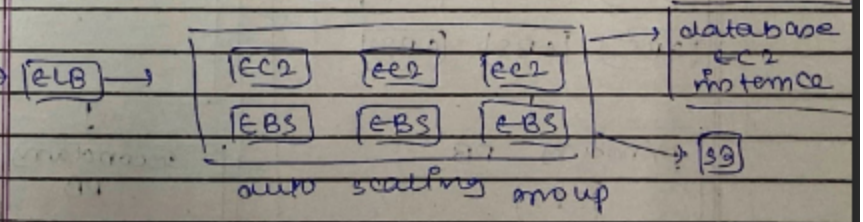
- initial deployment architecture:



→ a single ec2 instance is used as a starting setup to host application.
→ the ec2 instance host both web server and database server.
→ EBS - root volume of ec2.
- safeguards data in case ec2 fails.
- stores application code and database files.
→ S3 - used for storing and serving static content such as audio and video files.

for non business initial app.

- intermediate plain with auto scaling (dynamic scaling)



→ auto scaling can be used if application faces traffic load entire days.
→ also provides backup mechanism if a ec2 instance fails as a new instance would immediately be started.

→ Here, ELB (elastic load balancer) is used to balance load evenly among different instances.

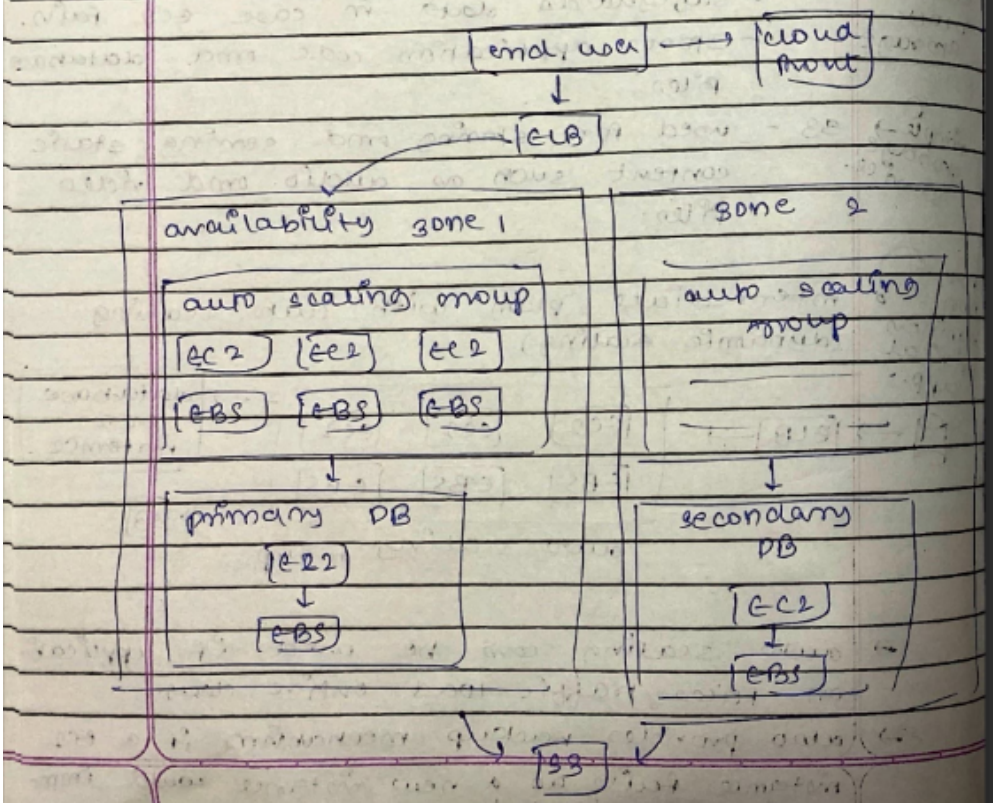
→ provisions resources according to the demand.

→ seperates database instance.

→ It faces both peak traffic and low traffic.

• Advance plan with high availability

→ used for business critical applications where no downtime can be tolerated.



availability zone

• adds high availability

• If a instance fails, application continues to be operational with slightly reduced performance. and this is done using another availability zone

• cloud front is used to cache static data closer to the users.

• all availability zones must have identical configuration to ensure proper reliability.