Topics	understanding			
what is network and internet	1. Network It is a collection of devices used to communicate from one device to another. 2. Internet It is the medium to connect networks to devices, which is used to access resources in a network or a collection of networks.			
how internet works	Workflow **Phone → Cellular Data → Tower or Router → Optical Cable → Data Center** Data Flow **Data → Packet → (6-bit [0 or 1]) → Protocols → Destination** A normal user uses a mobile device to access data or any website or resources that need a common network. Most people connect using mobile network cellular data, which provides access to the internet. Then, the user types the domain name or destination address. The domain name maps to a specific IP address using DNS (Domain Name System), which maps IP to domain. Then the IP is identified, and it searches among servers to locate the resource. Finally, the result is displayed as a response to the user's mobile			
compute and computer	1**.compute:**A set of tasks to be performed to give output. Example: Basic compute devices were built for arithmetic operations; later, they were upgraded to modern devices called computers. 2.A **computer** is a combination of hardware and software. It is an electronic device used to perform a set of tasks or instructions based on user inputs and provides			
basics components of computer	Components: 1.I/O Devices 2.Memory → (RAM, ROM) 3.NIC 4.CPU **RAM (Random Access Memory)** Used to store and process temporary data. It can store items in different and anywhere locations in the RAM. **ROM (Read-Only Memory)** Used to store and retrieve data in a structured manner. The locations are predefined. **SSD vs HDD** SSD: Modern storage device that uses advanced read and write mechanisms to easily locate the desired location. HDD: Uses a needle to locate the desired location to traverse data for read and write operations. **MAC (Media Access Control)** A unique identifier for a device to transmit data over a network.			
server	**Server** A computer used to share resources among various devices through the network. **Types of Servers:** Email Server Web Server Storage Server **Normal Processor vs Xeon Processor** Normal processors like i7 or N series do not support ECC (Error Correcting Code). **RAID (Redundant Array of Independent Disks)** Used to replicate data across disks. It consists of various types like RAID 1, RAID 2, etc. **Types of OS** We can create any type of OS server like Windows, Linux, Ubuntu. **Storage Server Port: 3306 for MySQL.** **Client and Server Interaction** Client requests data from the server; the server processes the request and sends a response to the client. **How Data Centers Are Maintained** Constant power supply Location of servers Internet connectivity Security Management engineers Cooling systems			

computer network	**Components of Network** Network devices Network interface Network medium Routing devices Connecting two or more devices using **a switch for wired or a router for wireless.** **Layers of Network** Application → Presentation → Session → Transport → Network → Data Link → Physical Based on the layer, the data is formatted and transferred from one device to another			
types of networks	**Types of Networks** 1.LAN (Local Area Network): Used within a single building like home, office, hospital. 2.PAN (Personal Area Network): Used to share connections between devices within 5–10 meters (Bluetooth, NFC, USB). 3.CAN (Campus Area Network): Connects multiple LANs (e.g., inside Anna University, many blocks are connected). 4.MAN (Metropolitan Area Network): Connects various metropolitan organizations based on geographical location. 5.WAN (Wide Area Network): Connects networks across the world, combining multiple types of networks. 6.SAN (Storage Area Network): Used to access storage among the network without traffic.			
Topics	understanding			
networking devices	**HUB** It is used to transmit or receive the data to all the devices in the network. Example: When a network consists of 4 devices, Device 1 sends data that needs to go to Device 4, but it transmits to all the devices present in the network. It may cause traffic. **Switch** It is used to send the data from source device to desired destination device. Example: When D1 sends data to D2, it only travels through the path and sends it. It reduces traffic and bandwidth. It separately maintains an ARP table. **ARP** Address Resolution Protocol, used to map or store the MAC address with the repeated IP in the ARP table. **Router** When the network device needs internet, the router will provide internet and IP to all the network devices. Used to connect to the outer world internet. It can be private or public network. **Repeater** Acts as an extender, used to make a copy of the signal and transmit or make connection to extra distance to access the network. **Bridge** It acts as a join or way to connect or access different networks. **Wireless AP** When the network connection is oriented in wired with cable, and we need to connect multiple devices, at that time we make it wireless using a wireless access point. Simply, WAP is used to make a wired connection of IP or internet into a wireless connection establishment. **Switch,Hub**->create the network **Router**->Connect the network			

Ports and protocols	**Port** A port acts as an endpoint to exchange the data. Example: When a user needs resources of a website, the port follows IP, then it displays or gives the output website data. Here we get the data, fill the form, so it makes exchange of the data. **Types:** Virtual Port: Ports used to access software, web applications, programs running. Physical Port: Ports used for input to computer and output from the computer to share data, like USB port for pen drive to exchange data. **Network Ports:** **Well-known ports**: Used for network communication properly, such as HTTP, HTTPS (80, 443). **Registered ports:** Assigned to specific organizations for service requests that provide necessary data or access to service. **Dynamic ports:** Not assigned to organization service, mostly private or temporary purpose. **Protocols** The set of rules or principles needed to follow for the flow of resources to be shared. In modern data transmission, data sent to a device is divided into packets and sent one by one in desired sequence or manner, then collected and formed into the original data for use. **Common Ports and Protocols:** 22 → SSH: Used for remote login for Linux OS or VM. 80 → HTTP: Hypertext Transfer Protocol. 443 → HTTPS: Secure connection that encrypts the data transmitting between client and servers. 20, 21 → FTP: Transfer files from one device to another through the network. 67, 68 → DHCP: Dynamic Host Configuration Protocol; when the device connects to a new network, it automatically assigns IP for each device.			
Topics	understanding			
workflow of internet work	**1. Network Flow** :Client Request When you enter a domain name (e.g., www.example.com) into your browser, the browser needs to find the corresponding IP address of that domain to communicate with the correct web server. **2. DNS (Domain Name System)** DNS is used to map domain names to IP addresses. Humans find it easier to remember names (www.google.com) than numbers (142.250.72.14). DNS translates the domain name into its matching IP address. **3. ISP DNS:** ServerYour request first goes to your ISP (Internet Service Provider)'s DNS server. If the ISP's cache already has the IP address, it sends it back immediately. If not, it forwards the request to higher-level DNS servers. **4. Root DNS Server** If the ISP DNS doesn't have the record, the query is sent to a Root DNS Server. The root server doesn't have the IP itself, but it directs the request to the correct Top-Level Domain (TLD) server based on the domain extension (like .com, .org, .in, etc.). **5. TLD (Top-Level Domain) DNS Server** The TLD server manages domains with specific extensions. For example, if the domain is example.com, the query goes to the .com TLD server. This server then points to the Authoritative Name Server for that specific domain. **6. Authoritative Name Server** The Authoritative DNS Server holds the actual DNS records for the domain. It provides the final IP address of the website. This IP is sent back through the chain (TLD → Root → ISP → Client). **7. Network Device Connection** Once the IP is known, your device connects to the server through a series of network devices: Device → Wireless Access Point (WAP) → Switch → Router → Internet Gateway → ISP	4		
Topics	Explanation			

http,https,ssl,tls	HTTP – Hypertext Transfer Protocol. It is used to exchange text or hypertext data from client to server and server to client. It is an application-layer protocol. It sends the data as actual text, making it vulnerable to data security issues, as any third party can easily see the data. Port: 80 HTTPS – Hypertext Transfer Protocol Secure. It is an updated version of HTTP. The data transferred from client to server is encrypted, and decryption is required to read it. To achieve HTTPS, SSL or TLS certificates must be configured. Port: 443 SSL – Secure Sockets Layer. It uses SSL certificates to create a secure tunnel for exchanging data between source and destination. SSL uses older cryptographic algorithms. TLS – Transport Layer Security. It uses updated cryptographic algorithms and keys to exchange data securely. TLS is an updated version of SSL. Note: Forward Secrecy: Even if the server's private key is compromised in the future, past			
Subnet and Subnet mask	Subnet – If a network provides access to another network outside the organization, the outer network can potentially access devices inside the network, creating vulnerabilities. To prevent this, subnets are used. A subnet is a subset of a network that divides the network into multiple smaller networks. Octet – The IP address is divided into octets, e.g., 10.0.0.0 → octet.octet.octet.octet (8-bit binary per octet). How to convert or assign subnet IPs: Determine the default subnet (based on the class of IP), e.g., 255.255.255.0 Find the CIDR (Classless Inter-Domain Routing) notation. Borrow the required number of 1s for subnetting.			
Topics	Explanation			
Ping	**Ping:** Ping is the command used in command line; it is used to troubleshoot issues between source device and destination device. It first sends 4 data packets to the destination and receives them back. Based on the response, it gives output like "Request timed out" or "Packets received." Based on this, we can identify where the problem is. Syntax: ping IP or domain name			
tracert	**Tracert:** Tracert is also a command used in CMD. It is used to trace the hops and identify which device has a problem, and what intermediate routers or devices transmit the request to the destination, along with the response time. It lists all device IPs. If there is any problem on connecting devices, it is used to identify it easily. Syntax: tracert domain name or IP these above both use icmp. note:traceroute (Linux/macOS) and tracert (Windows).			
telnet	**Telnet:** Telnet stands for Telecommunication Network. It is a CLI application used to connect to resources anywhere; in other words, it is used to remotely access devices, files, folders, applications, operating systems, virtual machines, storage, etc. It does not encrypt the data; it transmits data in plain text and actual format. All the output will be CLI application. It uses port 23. Syntax: telnet hostname or domain_name portno			

ssh	**SSH:** SSH is the protocol and CLI client used to remotely access all files and VMs, same as Telnet. The data transferred from source to destination is always encrypted. It uses port 22 and is used to protect resources with passwords and keys. Syntax: ssh username@[hostname or IP] port or ssh -i path_to_key username@[IP or hostname] port To end the connection, use the exit keyword or Ctrl+C.			
Topics	Explanation			
Proxy	Proxy is the server, the server that present in between client and server, that masks the IP of client and sends proxy IP to main server, then gets response from main server, stores the copy of the page and then sends back to client. It helps to achieve the privacy; the main server does not know the actual client. Based on cache it retrieves the data or webpage in speed. It also has the logging of which client accesses which server. There are two types of proxy. Forward proxy: the forward proxy is what actual proxy. Reverse proxy: the proxy stands in front of server that has the configuration of IPs of server that what should do it based on the condition it specified, and it routes to server.			
vpn	**VPN** stands for virtual private network, it is used to connect the device or application from anywhere, using private IP tunnel, which creates private IP and that config to that root device. By using the tunnel it transmits the data, while request–response process it not trackable by any other authorized and unauthorized organization. connection types:site -network,point-devices **point to point** **site to site** **point to site** **site to point** actual types: **half tunel**:when the connection establed between two device when A and B ,the device A need to acces internet ,the a need the INternet router to acess ,the connection betwwen b is only used to share the data purpose **full tunnel.**:full tunel is used to acess the internet throhgrn the device b it also acess internt from that connectio			
Topics	Explanation			
OSI layer-part-1	**OSI Layer:** NOTE:Open Systems Interconnection. It is a model used in computer networking to explain how data moves from one device to another across a network. Application Layer Presentation Layer Session Layer Transport Layer Network Layer Data Link Layer Physical Layer			
Application Layer	Application Layer: It is the topmost layer of OSI. The layer represents the user interface of the application. It gets the interaction of the user or user inputs from any device that handles the actions triggered by the user.			

Presentation Layer:	Presentation Layer: The layer is the second layer from the top. It is responsible for encryption, compression, formatting, and also represents the incoming data from one format to another. For example, the data comes in the format of 0s and 1s; it identifies the correct format to which it belongs, like image, video, or file. Based on that, it formats that type to the actual type to show to the user.			
Session Layer:	Session Layer: It is the layer that establishes the session for resources to access. It sets a time to keep our data; when the tab of a site or resource gets closed, its session ends. It also provides user-defined management to make applications work correctly.			
Topics	Explanation			
firewall	**Firewall** The firewall that protects and stops all the incoming traffic, and allows the access to specified IP networks to access the data inside of the organization and outside of the organization. In other words, it allows and manages inbound and outbound rules based on user-defined settings. The firewall stops and allows the permission for each request based on Port, Protocols, Programs, Domain, IP, Allow/Deny. **There are three types of setting up the firewall:** **Stand-alone firewall** - It is physically present on premises. **Cloud firewall** - It can be done on any cloud service provider like AWS or Azure to protect the resources available on the cloud or on premises. **Switch firewall** - The switch has a dedicated integrated firewall to manage incoming and outgoing traffic. Types: **Network-based firewall:** The firewall that is present at the entry point of the organization. **Host-based firewall:** This firewall is software-based; it protects the individual devices of the organization.			
DMZ	DMZ stands for Demilitarized Zone, it's also known as a Perimeter Network. The DMZ is the structure that keeps the exposed servers, databases, or VMs that are frequently used by the public, and it is all available over the public internet. It is kept outside of the main organizational firewall and set with a separate firewall to access the exposed servers or services. It is also possible to implement a DMZ using a home router to allow specific IPs in the router configuration. Example: PlayStation at home to access all the connections. **Let's say, for example**, that there is a startup company that hosts an app to identify whether the data or AI-generated content is real. The main servers are inside the organization. Most people request and access the app server, but all internal servers and devices are connected to the same network behind the firewall. This can cause any damage and possible hacking of other devices. To avoid that, we use a DMZ structure, which keeps the app server in front of the main firewall and places another firewall in front of the app server. This helps to make the network secure.			
Topics	Expliation			

OSI layer-part-2	**Transport Layer:** The transport layer divides the data into respected bits and bits into segments. The segment contains the header + data unit, source port, and protocol. This also contains flag, ACK, SYN connection. The main transport layer protocol is TCP, UDP. **Network Layer:** The network assigns the IP for the segment which creates a packet. The packet contains source IP and destination IP, with version of IP like IPv4 and IPv6. **Data Link Layer:** The data link layer contains frames. The frames contain MAC address of devices and destination MAC. Here destination MAC defines the nearby router. Based on network data the router gets hop and hop to reach the destination IP and get the MAC and deliver the data based on the MAC address. **Physical Layer:** It is the medium that data gets transferred. There are two media: wired and wireless. For wired we use Ethernet cable; for wireless we use access point, routers. It gets transferred into electric signals. The signals contain the 0 or 1.			
DHCP Server	The **DHCP** is used to assign the IP for the device inside the network. This follows the DORA process. When a new device needs to join the network it needs to undergo the process: D – Discover: The new device broadcasts the MAC address of the device and finds the DHCP server. O – Offer: The DHCP receives the MAC and offers the IP to the device. R – Request: The device sends back the request that it has configured the IP. A – Acknowledgment: At last, the DHCP sends the acknowledgment that keeps the MAC and assigns the IP.			
FTP Server	The server is responsible to share the file from one device to another. It stores the file data. It is dedicated for file sharing.private,public ftp server.,port 20,21			
Topics	Expliation			
Active directory	**Active Directory (AD)** is a service provided by Microsoft, specified in Windows Server operating systems. It helps organizations list, organize, and maintain their IT systems efficiently. **Active Directory has three main components:** **Users/Resources** – Represents the individual accounts and resources within the organization. **Groups** – Used to manage users collectively based on roles or purposes, such as the HR team or the Cloud team. **Domain** – Controlled by a Domain Controller (DC), which manages authentication, authorization, and access control. When access needs to be granted to a set of users, credentials are provided, and authentication and authorization processes are applied. Groups are formed to achieve common goals, such as departmental collaboration or project-based access. **Organizational Units (OUs)** An Organizational Unit (OU) is a collection of users, devices, or both. OUs are used to manage resources more efficiently. For example, you can assign Group Policy Objects (GPOs) to OUs. A GPO is a set of rules or restrictions that can be applied to specific users, devices, or OUs. For instance, if the organization wants to block firewall access on all devices within a department, a GPO can be assigned to the corresponding OU to enforce this policy. Domain Structure in Active Directory **Active Directory uses a hierarchical structure:** **Forest** – The top-level structure, also called the Forest Root Domain. It contains all domains, trees, and OUs within an organization. **Tree** – A collection of domains under the forest, usually organized by region, branch,			

Topics	Explanation			
Application Artitecthure	**Single-Tier Architecture:**The application consists of three main components: Frontend, Backend, and Database, all wrapped into a single layer or system. Pros: Easy to manage, cost-effective. Cons: Data loss if server fails, difficult to debug.			The application consists of three main components: Frontend, Backend, and Database, all wrapped into a single layer or system. he architecture consists of two parts: 1. Client (UI + Server) 2. Database Here, the database server is separate. Divides the application into three layers: 1.
	Two-Tier Architecture: The architecture consists of two parts: 1. Client (UI + Server) 2. Database Here, the database server is separate. Pros: Data recovery, replication, scalability of database. Cons: UI and server are tightly coupled.			
	Three-Tier Architecture: Divides the application into three layers: 1. Presentation Layer (UI) 2. Application Layer (Backend/API) 3. Data Layer (Database) Pros: Easy to manage, better failure handling. Cons: Higher cost due to separate servers.			
	N-Tier Architecture: Extended version of three-tier architecture with multiple layers . Pros: Highly scalable, modular. Cons: Complex, requires more servers and infrastructure.			
	Microservices Architecture: Breaks the application into independent services (modules) that handle specific functions. Each microservice runs separately and communicates via APIs. Pros: Fault isolation, scalability, flexibility. Cons: Complex deployment and monitoring.			
Topics	Explanation			
Enterprise Architecture	Enterprise Architecture is a framework used to build business and logic-based applications that represent the components of an organization and the resources utilized both inside and outside the organization. It consists of various layers and can be implemented using one-tier, two-tier, or n-tier architecture in the application layer. Example: Consider a product-based company. The company needs internal portals such as HRM (Human Resource Management), CRM (Customer Relationship Management), and Finance portals to be maintained within the organization. At the same time, the main application that monitors the product should be accessible externally via the internet as the company's application. To achieve this, the organization must create a network based on its business requirements and manage it effectively.			
Topics	Explanation			

Virtualizaton	**Virtualization** Virtualization is the process of manipulating computer hardware resources to run multiple operating systems (OS) on a single physical device. The software used to create and manage virtualization is called a hypervisor. **Types of Hypervisors** **Type I Hypervisor (Bare Metal)** Installed directly on the physical hardware without any host OS. Helps create and manage multiple guest operating systems. Example: VMware ESXi. **Type II Hypervisor** Installed on top of an existing host operating system. Allows running multiple virtual machines above the host OS. Examples: Oracle VirtualBox, VMware Workstation. **Advantages of Virtualization** Saves money on electricity and hardware components. Reduces physical space requirements. Enables testing and working on multiple operating systems easily. **Disadvantages** If the host OS fails, all virtual machines are affected. Lower performance and slower boot times compared to physical machines.			
evolution of virtualization	Gen 1: Physical machines – actual computers running one OS with multiple applications. Gen 2: Virtual machines – multiple OS instances running on a single physical machine using virtualization. Gen 3: Containers – lightweight, portable environments for applications.			
Container	**Containers** package application code, configuration, dependencies, and requirements into a single unit called an **image.** When an image runs, it becomes a container, which executes on a container engine (e.g., Docker). Compared to virtualization, containers are lightweight and faster. Applications in containers are portable and can be shared or deployed anywhere.			
Topics	Explanation			
cloud computing	**Cloud computing:** A set of computer resources shared over the internet and accessible from anywhere in the world by remote access, using a pay-as-you-go model, which means you only pay for the utilization of services as required.			
how cloud transformation from on-prem?	**how cloud transformation from on-prem?** When there is a set of physical computer resources available, and we need to deploy services, we have to go to the physical location and then deploy the application or servers. When it is moved to the cloud, we can access these data centers from anywhere through the internet. It can be accessible for public and private use.			
cloud service provider	The major service providers are Oracle, IBM, Salesforce, Azure, AWS, Google.			

cloud deployment models	**Public Cloud:** The cloud resources or applications, servers, all that are available to the public. **Private Cloud:** The computer resources, applications, servers that are available only within the network, not to the public. It can be achieved from a private data center owned and managed by the organization. **Hybrid Cloud:** The combination of public and private cloud. The computer resources are available both in private and public networks based on requirements. **Example:** When the client needs to connect the on-premises network to the cloud environment, that is the best example of a hybrid model. Why hybrid? When the on-premises servers are not capable of managing traffic or need extra servers but have no space on-premises, we use a hybrid model. Also, for security purposes: for public users, servers keep the data available in the network, and some confidential servers and applications remain inside the private network.			
cloud services model	**Cloud services model:**The cloud provides various service or delivery models for our convenience to utilize the services. **IaaS (Infrastructure as a Service):** IaaS provides overall computer resources like empty servers, storage, RAM, all things without any configuration. How: When we need to control computer, we need an OS. For that, create a VM and then configure required things. **Examples:** Microsoft Azure VMs, AWS EC2, Google Compute Engine. **PaaS (Platform as a Service):** The services have default setup or auto configuration for storage, network, OS, etc. It requires the application and data to be hosted on the internet. The application is automatically deployed without any manual configuration. It's easy to deploy the application. Examples: Azure App Service, Google App Engine, AWS. **SaaS (Software as a Service):** The software that is available and ready to use over the internet for the public. **Examples:** Microsoft 365, Email, Teams.			
Operation	**Deployment/Config:** The deployment is the process of installing the application or set of code into a cloud. The configuration is needed for the deployment process to support. **Patch Management:** The process of fixing the bugs, or if any features are not working after deployment, we give a patch update to fix it. These kinds of processes are in patch management. **Change Management:** The change management is the process to notify or address the action or commit to all the people involved in the project. **Example:** If one person made a change and did not announce it to all, but all other developers are working on the old version of code, it goes into final deployment and that makes conflict on the whole application. To avoid this, follow the change management. **Incident Management:** The log of activity or steps that are done when the problem occurred, what steps to follow and how the problem gets solved. Note these are data in log type, like cause, solution, output. **Break-Fix / Hard-Fix:** It is the action that is taken on deployment stage, when we need to stop the application in minimum amount of time and when we know the root cause of the problem and have the solution to solve it, then in that case we use the hard fix method to resolve the problem in deployment.			
Topics	Explanation			

Operation	**Automation** The automation is the process of automating things without any human effort or manual intervention. When we perform an action, the respective action needs to meet a process and complete. **Example:** When a developer makes a change in code and gets it updated, after that update the code needs to create the updated image and deploy it into the respective server. **Monitoring and Logging** **Monitoring:** The monitoring is the process at the infrastructure level that checks performance and sends notifications or triggers some services based on requirements. **Logging:** Logging is the process of listing the activities that have been done in the application or cloud-level activity. Based on logging, we can identify who, when, and what activity was done. **Performance Tuning** It is the process of increasing or decreasing the computer resources based on the history of metrics. **Example**: When the application is specified with specific resources like 8 GB, and the application is only using 2 GB, we can reduce it. This is known as performance tuning.			
open sources,freeware and freemium diffrences	The **open source** is the software or OS or any function that can be modified by the user for therse convenient and configurations and used. The **freeware** cannot be modified by the user, only used as software. The **freemium** refers to some part of the application being available and accessible for free, and the main part of the application cannot be accessed without paying an amount.			
os	The operating system is communication between hardware and software that manages storage, RAM, processes, and all necessary components to run applications.			
BIOS & boot process	**Basic Input/Output Devices:** It is responsible for connected devices and the initial boot process. When the button is clicked, the BIOS starts and makes **POST** (power -on-self- test)to verify the I/O devices, then it searches for OS files that are present inside the MBR (Master Boot Record). The **MBR** is located in the first 32 sectors, that have 3 parts, which divide 512 bytes. The 1st part consists of boot loader(code to start os), partition tables((info about disk partitions), and boot signature. When the MBR starts, the path to the file that loads the OS file into RAM is set, and the OS gets started and runs. The MBR only attaches until 2TB and is limited to 4 partitions. **UEFI:** It is the modern firmware that replaces the BIOS in modern devices and new generation systems. Like MBR, it uses GPT, which stands for GUID Partition Table, that has no limit for storage and partitions. is limited to 128 partitions.			
Topics	Explanation			
CMOS	The CMOS is non-volatile memory that plays a crucial role in BIOS. This is a separate component that is embedded with the motherboard. It stores the setting configuration to load into the BIOS. The CMOS is also responsible for retaining memory after turning off the computer. It is also responsible for time and date.			

boot process in Linux	**Power On** When the power button is pressed, the BIOS (Basic Input/Output System) starts. **POST (Power-On Self-Test)** BIOS performs POST to verify connected I/O devices (keyboard, mouse, storage, etc.) and ensure hardware is functioning properly. **Search for OS Files** After POST, BIOS looks for operating system files inside the MBR (Master Boot Record). **Locate MBR** The MBR is located in the first 32 sectors of the storage device. It is 512 bytes and divided into 3 parts: Boot code – Instructions to start the OS. Partition table – Information about disk partitions. Boot signature – Validates the MBR. **Load Bootloader (GRUB)** The MBR loads GRUB (Grand Unified Bootloader). **Execute Kernel** GRUB executes the kernel, which is the core of the operating system. **Initialize OS** The kernel initializes the OS and sets the required run level: 0 – Halt (system off) 1 – Single-user mode			
Windows Artitechture	**The Windows architecture is divided into 4 layers:** First layer: Hardware layer – This layer consists of physical components like RAM, ROM, BIOS, motherboard, GPU, etc. **Layer 2 consists of 3 parts:** **Bootloader** – It is responsible for loading the OS into RAM. **Board Support Package** – Helps to run the BIOS and all necessary programs to start and run at the hardware level like BIOS. **Device drivers** – When a new device is connected to the system like mouse, pen drive, keyboard, we need programs to access and communicate with these devices. For that, we need drivers. Mostly all necessary drivers are already installed; if not, they can be installed manually. **Layer 3** consists of the actual **OS** that helps to interact and communicate between software and hardware. This layer has three parts: File system – It is the protocol used to store and retrieve data in a specific manner, like FAT32. GUI – Stands for Graphical User Interface that is visible to the user. By GUI icons, we click and drag to access software and applications. Task management – The process of monitoring and managing all tasks running when the OS is on. **Layer 4: Application layer –** The application layer contains software, web applications, and all programs.			
Topics	Explanation			
Patch & upgrade	Patch: A patch is a set of code used to fix a bug or problem in an already existing application. It is installed over the already installed application. Upgrade: An upgrade is a large set of code or a new version of the software. When an update is needed, the old version of the software is uninstalled and the new version is installed.			

Artitechture of linux	**The main component in Linux is the kernel.** The kernel is used to communicate with hardware. The kernel is completely editable; we can define and install modules. The terminal uses an interpreter that communicates with the shell. Linux supports GUI with minimal installation. **Linux architecture contains 4 layers:** Layer 1 – Hardware: Where the actual physical components are present, like CPU, RAM, ROM, BIOS, CMOS, etc. Layer 2 – Kernel: The kernel directly interacts with hardware and performs OS-level operations. Layer 3 – Application Layer: Presented on top of the OS, it contains libraries and system daemons responsible for task management. The shell is used to communicate between the user and the OS. Layer 4 – Tools: Tools are installed in the OS to manage it, like firewall, SSH tools, etc.			
Topics	Explanation			
Linux	Linux is the open-source operating system. Initially, the OS was under licenses and cost to use UNIX. After that, one person created the OS for their need with the distribution of UNIX named as Linux. After the OS was made open source, then most users used it and modified the OS as users' needs. Then it came into more distribution with the different aspects of Linux. Then it was made open source of Linux **distribution** like Ubuntu, Kali. Inside each distribution have different flavors, like Ubuntu has different flavors such as Edubuntu, Kubuntu, Ubuntu Studio. **Flavors** are built based on specific process improvement.			
Linux directory Path	The Linux has File System Hierarchy, the important path in structure is: **Relative path:** The specific path that is inside the current directory or inside its root directory. Example: rizi@user/User/home; cd rizi/file1 **Absolute path:** The path starts with '/', that path consists of the whole root path to reach the files or folder in directory. Example: rizi@user/User; cd /home/rizi/file1/			

Linux directory Structure	**Linux Directories:** **/bin** – The file contains the binary code for the commands that are used in the shell or terminal. These are essential to use the commands to run. **/boot** – The boot file contains actual OS bootloader files, OS image, necessary files to load the OS. **/dev** – The dev stands for devices that are connected to hardware in the context of storage. The connected disk storage is listed here. To use and access these disks we need to configure the file system by mount process. **/etc** – The etc contains the system configuration, like change in username, device setting, like control panel in Linux. **/home** – The home contains the directory of the user, list of directories and files that are stored in the device for the specific user. **/lib** – The libraries contain the external or support dependency to run the code or application inside the Linux. This is also called shared libraries. **/mnt –** The mount contains the disks that are configured and mounted are listed here, to access the file and data from the disk. **/media** – The media contains the unmounted disks that already have the file system and temporary storage devices like pendrives, USB attached hard disk. **/opt** – The opt stands for optional. This file contains the packages that need to run and install the add-on application software packages. **/sbin** – Like bin, the sbin stands for system binary that is used for essential system required code like fdisk, ipconfig, mount and unmount, that are used to interact with system directly. **/srv** – Services based data. While using site, the site-based data gets stored on this srv with separate directory. **/temp** – The temporary file that contains all the running application created data temporarily that are stored in temp. **/usr** – The user directory contains the list of user-specified settings that are used for specific user.			
Topics	Explanation			
Navigation Command	**Basic Linux Commands** **cd** – it stands for change directory. Used to locate the path of file or directory we need to access. We can give absolute path or relative path. Syntax: cd relative_path /absolute_path **ls** – the list (ls) is used to list the files and directories present in the current directory. Syntax: **ls -al** -a (show all the hidden files) -l (show the detailed information about the files and directory) **pwd** – present working directory, this shows which directory you are in now. **mkdir** – make the directory, it is used to create a directory. Syntax: mkdir folder_name -r (recursive process) -f (force to do) **cp** – copy, it is used to copy the file from source location to destination. Syntax: cp [-r] source_file/folder1 source_file/folder2 source_file/foldern Destination_file_location **tree** – the tree is the third-party tool that is used to view the file structure in tree structure. ***** – while deleting or copying files we can mark all using this * (means select all). **rm** – remove, it is used to remove the file, directory, or anything. Syntax:			

Linux Login Process	Login Process Steps 1.Login terminal needs to show, from the /etc/rc.local, the file executed and trigger action. 2.The action triggers the terminal that getty (it stands for get the terminal) and shows to user to enter the password prompt. 3.The password entered after it checks and compares the file /etc/passwd. If correct go to step 4 else step 1. 4.The user gets logged in to terminal and shows the message of the day, that is executed by /etc/motd. Example: we can modify that script to show the greeting and what are the functions need to run and show. 5.Then the getty reads the configuration for the respected user from the profile which is present inside the file /etc/profile and imports the file. 6.The profile loads the necessary functions that are present inside the scripts of bashrc and bash_profile.			
File Manipulation Command	**Create:** nano: it is a tool to create the file and popup the editor to enter text. To use it, after typing press Ctrl+X, then Y, then Enter. vim: it is also the same tool that has high steps to edit and save the file for security. For typing press i (to enter into insert mode), then Esc to exit from update, then type :wq (save and quit), :q (quit without saving). touch: used to create the empty file with any extension or none. **Read:** cat: used to display the content inside the file. **Update:** use same as create command; if file exists it opens or creates new. **Delete:** rm: remove the file from the directory or remove the directory. Hidden file: to hide the file from directory or normal ls command not get listed that kind of file. To find file from any directory globoly use **which file/folder_name**			
Topics	Explanation			
Chown & Chmod	**chown** stands for change ownership. It's the command used to change the ownership of a file or directory. Syntax: **chown user_name:group_name file_name** **chmod** stands for change mode. It is used to modify the permissions for operations on a file, such as: Read (r) Write (w) Execute (x) **Syntax:** **chmod NNN file_name** N = cumulative total of binary representation of operations. If permission is provided, put 1; else 0. For example: rwx → binary 111 → decimal 7 (4+2+1) **Note:** When using these commands on a file or directory, you need to be the owner of the respective directory or file, or you need to be a sudo or root user.			

User Administrative Commands	User Management useradd – Adds a user. To create a home directory for the user, include -m. Example: useradd -m user_name usermod – Used to give permissions or privileges to the user. You can assign the group to the user, e.g., give sudo access: Example: usermod -ag sudo user_name userdel – Deletes the user only; it does not delete the user’s home directory. lslogin – Displays the logs of users logged into the VM. groupadd – Creates a group. groupmod – Modifies a group (name, group ID, etc.). gpasswd – Changes and manages the password of the group. gdel – Deletes the group. **Note:** When using these commands on a file or directory,you need to be a sudo or root user.			
Virtual Machine Networking	**VM Networking Types** **NAT** – NAT networking connects the VM to the host IP. When the VM requests resources from the internet, it masks the private IP with the host IP and forwards the request. Internet access is available. **Bridge Network** – Creates a connection with any other physical network device to access the internet and intranet. It assigns a separate IP from the physical device. Internet access is available. **Host-Only Network** – The host device creates the network and assigns IPs to virtual machines. VMs can communicate with each other and access the host OS remotely, but cannot access the internet. **SSH Access** When the VM has OpenSSH server installed, it can be accessed remotely via SSH. To install: sudo apt install openssh-server To connect: ssh user_name@ip_vm			
Topics	Explanation			
Package Management	The package management is the package that contains all the software and tools. By specifying the name of the tool followed by using the package manager, there are various package managers designed and developed by the distro. A package manager developed in one distro cannot be used or installed in another distro. The third-party package manager can be installed in any Linux distro. Ubuntu has snap and apt package managers. Syntax: sudo apt install tool_application_name yum – RedHat package manager apt – Debian package manager pacman – package manager zypper – package manager dnf – Dandified yum synaptic – package manager aptitude – package manager			
File System	The file system is the protocol that is used to store and retrieve the data from the hard disk. When the data needs to be transferred into a disk, it must have the filesystem. Based on retrieving and storing encryption algorithms, there are some file systems: ext4, ntfs, fat32, xfs.			

Drives/Partition	The drives are the actual physical drives with standard storage space. The partitions are the logical separation or division of drives. Each partition has a separate file system.			
logical Volume Mounting	LVM is a storage management system at the logical level that is able to extend and reduce the size of the logical volumes. There are various layers present. After that, the physical layer consists of actual hardware, then partitions. After the partition, convert to physical volume. Next, it is collectively used as a volume group. Next to that, all these are collectively called a logical group. Then the logical group has the option to specify the file system. We have several commands to do that below: pvcreate /dev/sdb – to create the physical volume vgcreate vg_name location_of_pv_partition – to create volume group lvcreate -L SIZE lv_name – to create logical volume To verify the pv, lv, vg, use: sudo pvs or sudo lvs or sudo vgs			
Topics	Explanation			
Install file system and mounting	After creating the logical volume, we need to mount it to use it. To mount, we need to specify the file system and the source of the logical volume and the destination directory of the mount point. Syntax: sudo mkfs.ext4 /dev/vgname/lvname sudo mount /dev/vgname/lvname /mnt/myvolume_name			
log management	The log is data that states the set of activity of a user to an application or server. It is used to identify the root causes of the problem. We can configure the log backups as date or month or year specified. logrotate – It is the config file that is present inside /etc/logrotate.conf. We can edit it. It consists of parameters: create log, rotate it, make remain when old logs get deleted. Then we can specify the size of the log; based on size it remains if it's small. When need to log, need to delete or trigger any notify, we can set it and also send email alert to user.			
demon	Daemon is the background process. We can start and stop and delete the process by using below commands: systemctl status service_name systemctl start service_name systemctl stop service_name			
cron Tab	Cron tab is used to perform a task in desired time as user defined. The task executes at time repeatedly until stopped. The cron tab is the daemon process that needs to run in background at all time. It starts when system gets started. Syntax: min / hour / day_of_month / month / day_of_week <command to execute> crontab -e :to edit cron AT: It is also a task scheduler. It contains the command in simple English. We can only make some time-based changes, not fully modified. It is used for any simple and not frequent task. Syntax: at 10 am tomorrow <commands to execute>			

firewalld	The firewalld is the daemon process of Linux system. It is used to stop all the incoming traffic and allow the outgoing traffic and incoming traffic as user defined. The firewalld is the software-based firewall (host-based). It has default multiple zones, commonly 9 different zones. The zone has set of inbound and outbound rules. The services are the set of rules or traffic that need to be maintained, like FTP, HTTP. These are predefined and we can use them and able to edit them. Next is rich rule – it’s fully customizable as user convenient. We can create the rule and make it service as our need. Commands: systemctl start firewalld sudo systemctl start firewalld sudo firewall-cmd --zone=public --list-all sudo firewall-cmd --zone=public --list-services sudo firewall-cmd --zone=public --add-http			

Topics	Explanation			
Continerzation	Containerization The containerization is process of combining all the necessary dependency to run application, application code and base OS to run application into single unit. We can easily setup and run the containers in very few steps compared to traditional setup. Running the application is slower.			
Docker	Docker Docker is the containerization tool that is used to interact with docker engine, to create, run, stop, push, pull the image of the application. The image consists of three main parts: Dockerfile Dockerfile is used to create the image. It sets code to set the base OS and necessary files to run an application and run command of the application. Docker Image When you use docker build command it creates the image by using Dockerfile. The image is created with ID and name. Container When you run the image the containers get created and establish the application with expose in specified port.			
Docker Demon and Artitechture	Docker Daemon The docker daemon is responsible to process the client inputs, based on that retrieve the images locally and pull from public registry and repository. The dockerd is daemon, it runs in background process.			
Registry and Repository	Registry The registry is collection of repository, used to store the images or any set of files as user needs and also store the artifact. The artifact is the image that has all necessary config to run the application.			

Docker network	Docker Networking The docker is able to create the virtual network, by default it uses bridge network. Other than that it consists of host, none, overlay network. Host Network: The host network gives the host IP to the container inside the host network. We can access the container using host IP. None Network: The none network is used to create the isolated network. It means it does not have IP for the container. It is used only within the container, when need to get into shell of container and interact with the application. Bridge Network: The bridge network creates its own network inside docker, that assigns IP to each container present in that network. Based on this the containers communicate with each other without exposing to any host or outside the network. When want to access the container application we need to port forward to view the application in our network or outside of network. Overlay Network: The overlay network is used to connect with multiple containers or docker networks. We can communicate from one docker network to another network. It achieves container orchestration.			
Docker File	Sample Dockerfile Commands FROM - BASE IMAGE OF OS WORKDIR - Need to specify the working directory COPY - Need to copy all the files from root directory to image RUN - To run the necessary command for application or to install the application dependency commands CMD - Final command to run the application EXPOSE - Port need to run the image			
Docker Commands	Basic Docker Container Commands Run a container docker run IMAGE Run interactively docker run -it IMAGE /bin/bash List running containers docker ps List all containers docker ps -a Start a stopped container docker start CONTAINER Stop a container docker stop CONTAINER Restart a container docker restart CONTAINER Remove a container docker rm CONTAINER View logs docker logs CONTAINER			

Topics	Explanation			
Docker Volume	Container Storage and Volumes The container stores the data inside the container. When we need to search for the data, we need to know where the data should be stored. The data are stored inside the volume. The volume is created anywhere directory inside the container. The volume is the storage that is used by the container. That storage based on location and access is divided into three types: Anonymous Volume – It locates any random location inside the container and creates it not human readable and accessible. Named Volume – We can easily locate the folder by defining name for that. If need to access the data it is not possible but we can point the data where it is and where need to store. Bind and Mount Volume (Host Volume) – The bind and mount create the directory in host devices and map or mount to docker container file directory. What the data it stores there is replicated to host devices. It can be human readable. Syntax: sudo docker volume create name_of_volume docker volume ls docker system df -v docker run --volume name_of_volume			
Container life cycle	Container Life Cycle The container life cycle is about how the container gets created and destroyed. There are several stages below: create – To create container from an image run – To run the container have created or image pause – The pause is used to stop the port exposing, that means the container is running but not expose the application. unpause – To resume the paused container we use unpause command start – To start the stopped or created container stop – To stop the running container, it temporarily stops the container delete – To remove the container or delete the container			
Docker swarm	Docker Swarm The docker swarm is the container orchestration tool, to create and manage master node and work node. It is used to communicate the container across the docker network, with overlay network. The docker swarm in the format of yml. It consists of two main parts: Service Discovery – Identify the resources utilization and available resources of specific application based on that we can know where the memory free to run the application. Scheduler – Based on the information by service discovery it has the location of next image need to containerized in which docker network. Compare to Kubernetes the configuration is very simple.			
Docker compose	Docker Compose The docker compose is yaml file, that consists set instruction that used to perform the task, or used to create the container. When two or more containers need to be start parallelly need to specify the image and cmd to run image. The yml file gets executed by below command: docker-compose up			
Topics	Explanation			

Kubernetes	Kubernetes is a container orchestration tool that helps manage containers across host machines. Let's take **where it is exactly used**. When you are deploying an application in a container on a host server, and when you need to **scale these applications**, for that we need to create another server and replicate these functions and contain it into the newly created server. When you create like that, it comes under **horizontal scaling**. Like this, with two servers means I can easily manage. If any changes are needed, first we need to update the first server and next we need to update the second server. When it comes to needing to have a vertical scale of size into 5 or 4, we are not able to manage it manually, like going to server 1, then going to server 2, then going to server 3. If any changes are needed, we need to update manually. This is not possible. For this kind of stuff we can overcome using **Kubernetes.**			
Kubernetes Artitechure	Kubernetes is a layer that forms across the host machines that are available. These are considered as worker nodes. The one master node is a virtual machine. It has the control for all the host machines under the master network. It controls the network under the container. Let's take its main component: **Kubernetes Pod.** A pod is nothing but an abstraction layer that is created above a group of containers or a single container. The pod triggers the container to run repeatedly. When the container gets stopped or gets down, the pod pushes the container to run repeatedly. When the container fails, it restores the container. Inside the pod the container never gets failed; it actually pushes the container to run. The pods are managed by the master node. We can create pods using a **mainfest YAML** script. The YAML script is used to create and run in the master node. if need to create or update the pod we need to create an new version of mainfest and then run on master node ,then **api services** help to verify if need the change based on older version if need change the services manager execute the script and create the pods based on the file.The master node gets it and processes it. The master node consists of four components. One main component is the API server and it communicates with **ETCD.** That is the database used to store where and how the existing code is present, like existing pod size, existing already run script—they are stored in that place. Based on that, the **service manager** is responsible to create a pod based on the script. If the pod is not presented inside the worker node, it gets executed to create the required number of pods. When it comes to scheduling, the **scheduler** knows where the worker computer has the efficient resource and free computer resource. Based on that, the scheduler sends to the specific worker node to create a pod. The pod gets managed inside the worker node. We cannot directly communicate to the pod. Inside the worker node, there are key components called **kubelet.** The kubelet distributes the traffic among the pods. It is			

SSH key generation	**Key-Based Authentication for SSH** When we need to log in to remote resources, we usually enter a password. Alternatively, we can use key-based authentication for better security and convenience. **Steps to Create SSH Keys** On the host machine, create the private and public keys using the command: ssh-keygen -t <algorithm_id> -c "comments_for_the_key" After executing the command, the keys are generated inside the .ssh folder. **Managing Multiple Keys** If multiple keys are needed for authentication, create a file and enter all the public keys or specify a single public key. The file should be named authorized_keys. **Configure SSH for Key-Based Authentication** Modify the authentication configuration in the sshd_config file (located in /etc/ssh/). Enable the key-based authentication option and specify the key path. Disable password-based authentication. Restart the SSH daemon so the changes take effect. **Login Using Private Key** Copy the private key to the user who needs to log in. Use the following syntax to log in: ssh -i private_key_file username@host_ip			
Topics	Explanation			
Kubernets components	While creating the YAML file, we need to specify the kinds below and create the pods: **Pod** – A pod is basically created when it is part of a deployment. **Services** – The service is the network created inside the worker node. **Ingress** – It is a standard network entry point. **ConfigMap** – The ConfigMap contains the necessary API keys or credentials in a secure way. It works like environment variables. These credentials are kept inside the master node when creating a pod, in a human-readable format. **Secret** – The secret is hashed or encrypted using the Base64 algorithm. It does not store actual text credentials. **Deployment** – The deployment is the kind used to create a pod. **StatefulSet** – The StatefulSet is used for applications or containers that need separate data access. The data inside the container will not be erased when the pod starts, restarts, or exits. **DaemonSet** – This kind ensures the container runs on all available hosts. It places the container on every worker node.			

Deployement	Deployment Configuration File
	The deployment has the configuration file for creating pods. It includes essential configurations like pod name, template to use, container image, where to pull it from, and container name.
	Deployemt.yaml
	apiversion: apps/v1
	kind: Deployment
	metadata:
	  name: deployment_name
	  namespace: specify_isolated_connection_name
	spec:
	  selector:
	    matchLabels:
	      app: name_of_pod
	  template:
	    metadata:
	      labels:
	        app: pod_to_use_for_config
	    spec:
	      containers:
	        - name: container_name
	          image: image_path_or_id
	          imagePullPolicy: [IfNotPresent/Never/Always]			

Topics	Explanation			
Kubernets Cluster management Commands	**command:** to apply the yaml file. **kubectl apply -f <path_to_manifest.yaml>** to dispaly the pods. **kubectl get pods /kubectl get pod <pod_name>** to describe the pods or deployemt or any kind. **kubectl describe pod/kinds <pod/kinds_name>** to give namespace for kinds **kubectl apply -f <path_to_manifest.yaml> -n <namespace>** to list the namespace **kubectl get ns** to get the pod on name space. **kubectl get pods -n < <namespace>** to create the namespace **kubectl create namespace <name_of_namespace>** to get the deployement or kind based name space **kubectl get <kind> <kind_name> -n <name_space>** to delet the name space. **kubectl delete [#optional <kinds> <kind_name>] -n <namespace>** to delete the deployement file **kubectl delete deployement <deployement_name>** **to logs the pods** **kubectl logs <kinds> pod_name**			
Topics	Explanation			

Services	**Services :** Services are used to create network connections between: 1.Pod to Pod 2.Node to Node 3.Inside the cluster network These types are specified in the YAML file under the spec property. If not specified, the default type is ClusterIP. **Types of Services** **ClusterIP** Provides an internal IP to each pod. The IP does not change when the pod restarts. Allows communication between pods inside the same node and across different nodes within the cluster. **NodePort** Assigns a port on each node to expose the service. Pods can be accessed internally and externally using <NodeIP>:<NodePort>. Useful for exposing applications running inside the cluster to external traffic. **LoadBalancer** Assigns a public IP or endpoint to the service. When requested, the LoadBalancer controller interacts with the cloud provider to create a load balancer instance. This enables external traffic to reach the pods through the load balancer. **Note:** **Service Network:** Services assign a static IP to pods or nodes. **Pod Network:** Pod IPs are not static. When a pod restarts, its IP changes. By default, pods get a private IP.			
ingrerss	**Ingress:** Ingress is a Kubernetes resource used to route traffic based on URLs.While services manage traffic for specific pods, they do not collectively manage traffic across multiple services. To achieve this, Ingress is used.Ingress typically c			
Topics	Explanation			
Kubernetes Auto scaling	**Scaling** is process of increasing or decreasing availability of resources or services. The scaling is achieved for under utilization and over provision we need to scale down, when over utilization and under provision need to scale up. We can scale the pods into two methods with automated config. **HPA** - Horizontal pod auto scaling **VPA** - Vertical pod auto scaling			
Horizontal Pod Auto scaling	**Horizontal Pod Auto scaling** The horizontal pod auto scaling is the process of scaling up or down the number of pods based on the memory or CPU utilization. We can fix the constraint that minimum pods and maximum pods need to reach, with condition of average utilization of CPU or memory. We need to specify it inside any pod creation YAML. We need to specify the **below parameters:** **min:No.of pods** **max:No.of pods** **average utilization: CPU/memory**			
Vertical Pod Auto scaling	**Vertical Pod Auto scaling** Vertical scaling is the process of increasing the single pod compute resources such as CPU, memory. Based on constraint we specified the minimum and maximum of memory or CPU or both. Based on the average utilization we give it scale up and scale down automatically. We can give the **constraint like below**: **minimum CPU:units in m** **maximum CPU:units in mi** **minimum memory:units in Ki/Mi/Gi** **maximum memory:** **average utilization:**			

Kubernetes volume	**Kubernetes Volume** The volume is used to store the data of pods. When we store the data inside the pod means when it restarts it is lost. When it comes to volume we can achieve storage persistence. To achieve this we need to follow these three entities: **Storage class:** It creates the connection between actual cloud service by authentication and provides name to access the volume. **Persistent volume:** When you specify the storage class and create the volume need to be accessed for one or more pods need, here only provide set of storage block size that predefined. Here also we can extend the storage. **Persistent volume claim:** When need to use the volume need to mount the volume to pod is done by here to use the volume. We need to claim it before use. We can extend it on YAML file. If want to access the volume it requests to PV and if existing it allocates volume by use PV. When if not existing the PV size, then it creates the new PV and then assign volume for PVC when there is storage class name get used. When it not used it not able to create the PV automatically, it only uses the existing PV only.			
Topics	Explanation			
Terraform	**Terraform** is a technology used to create cloud resources without manual interaction by using code. The code is written in HashiCorp Configuration Language (HCL). When we write the code and save it in a .tf file, we use the apply command to create the resources. **Where is it used?** When creating virtual machines or any cloud services. We can create manually, but if we need 50 machines or services, manual creation is not possible. In such cases, we use Terraform. When we work on a project and set up cloud architecture and networking using Terraform, if another client needs the same architecture, we can reuse the existing Terraform code. This provides reusability. **Terraform components:** **Cloud Interface:** The actual resources present in the cloud **State:** The state that contains the already configured resources and must match the cloud **Terraform File:** The .tf file that contains the code to create the services			
Topics	Explanation			
Terraform Commands	**Basic Commands** **init:**-Initializes the working directory and creates necessary hidden files for Terraform to operate. **plan**:-Compares the current state with the cloud provider and the updated configuration. Shows what changes will occur. **validate**:-Checks the configuration files for syntax errors and validates correctness. **apply**:-Applies the configuration. If the state matches the cloud, it creates or modifies resources. If not, it shows the differences and asks for approval to overwrite. **destroy**:-Destroys resources or services defined in the state and on the cloud. **import**:-When manual changes are made on the cloud, use terraform import with the resource ID or .tf file to update the state. **fmt** (auto format):-Formats the Terraform code for proper indentation and readability.			

To set enviroment variable	**Environment Commands** printenv Lists all environment variables. export PATH="path" Adds a directory to the system PATH.			
Modularization	**Modularization** Modularization means splitting the Terraform configuration into multiple modules. Benefits: Easier to identify errors. Simplifies management and updates. **Structure:** Divide .tf files based on actual resources. Create templates for resources. Define variables in a separate file. Use main.tf to control all templates, map variables, and run Terraform.			
Drift & State lock	**Drift** Drift is the variation between the Terraform state file and the actual cloud resources. Occurs when:Manual changes are made to resources created by Terraform. **State Lock** State lock prevents simultaneous changes to the state file during resource creation or updates. While locked, no other operations can apply changes. Can be manually locked/unlocked. Ensures consistency during deployments.			
Reclaim Policy & Access Mode	**Persistent Volume Access Modes:** RWO (ReadWriteOnce):Volume can be mounted as read-write by a single node. Multiple pods on the same node can access it ROX (ReadOnlyMany):Volume can be mounted as read-only by many nodes. RWX (ReadWriteMany):Volume can be mounted as read-write by many nodes. RWOP (ReadWriteOncePod):Volume can be mounted as read-write by a single pod. **Reclaim Policy:** Delete: When the PVC is deleted, Kubernetes deletes the PV and the underlying storage. Retain: When the PVC is deleted, the PV moves to “Released” and the underlying storage remains untouched. Recycle (deprecated): When the PVC is deleted, Kubernetes wipes the volume data and makes the PV available again.			

Topics	Explanation			
AWS Cloud Computing and AWS Overview	**Why Cloud Instead of On-Premises Servers?** If we want to create servers on-premises, we need to invest in CAPEX (Capital Expenditure) and OPEX (Operational Expenditure). After some period, if we no longer need the servers, the investment becomes a loss. In such cases, we can opt for cloud providers to avoid upfront costs and pay only for what we use.			

AWS	**AWS (Amazon Web Services)** AWS stands for Amazon Web Services, one of the most famous and widely used cloud service providers. It provides on-demand resources over the internet using a pay-as-you-go model. AWS offers around 240 services, continuously updating to make resource creation easier for users. **Cloud Migration to AWS** When migrating from another cloud to AWS: AWS often provides funding for migration for approved clients. Migration processing and engineering costs are funded by AWS. This is why most migrations happen with AWS support.			
AWS Infrastructure	**AWS Infrastructure** AWS provides data centers to store services and compute resources. Data Center = Availability Zone (AZ). A collection of 2–3 Availability Zones is called a Region. Regions are named based on country or state (e.g., us-east-1 for North Virginia). North Virginia is the only region with all AWS services (also where AWS was founded) **High Availability** We can replicate compute resources inside the same zone at no extra cost. This ensures high availability. Service availability differs by region; some services are region-specific. **Redundancy** Continuous replication of data from one device to another for fault tolerance.			
Disaster Recovery Strategies	**Disaster Recovery:** When the server goes down or any data center in the region becomes unavailable, the server gets down. To recover, we can use recovery management, which replicates the server based on different types of strategies **Pilot Light Strategy** Data is continuously replicated and actively running. Compute resources (like EC2) remain off until needed. **Warm Standby Strategy** Half of the compute resources are on and functional. Uses partial specifications compared to the actual server. **Backup and Restore Strategy** Only data is stored along with configuration details. Based on this data, servers and applications can be recreated when needed.			
RTO and RPO	**RTO (Recovery Time Objective):** How long it takes to restore a service after a failure. **Examples:** How quickly workloads come back online. How fast users regain access. How long applications remain unavailable during a disaster. **RPO (Recovery Point Objective):** How much data you can afford to lose. **Examples:** How far back in time your last usable data copy is. Data loss tolerance if systems fail.			
Topics	Explanation			

AWS Authentication	**AWS Authentication** Authentication is the process of verifying identity to access the AWS Management Console or AWS services and resources. It provides secure access to the AWS environment. There are two main types of authentication: IAM (Identity and Access Management) SSO (Single Sign-On)			
IAM	**IAM (Identity and Access Management)** IAM is used to manage users, groups, roles, and policies after an AWS account is created. **User:** A single identity with a username and password. Policies can be attached to define permissions. **Group:** A collection of users. Policies applied to a group affect all its users. **Policies**: A set of rules that define access to specific resources. Policies consist of statements (building blocks). At least one policy must be assigned when creating a user. **Roles:** Used to establish trust and allow communication between AWS services or external accounts (cross-account access). Roles rely on trust relationships.			
SSO	**SSO (Single Sign-On)** SSO allows a single set of credentials to authenticate across multiple applications or websites to access resources. **Requirements:** A domain and a Domain Controller to manage accounts. In the AWS console, permissions, users, and group policies can be managed centrally.			
AWS Organizations	**AWS Organizations** Organizations group multiple AWS accounts into a single entity for centralized billing and management. To enable SSO across all accounts in an organization: Use an Identity Provider (IdP) like Google or Microsoft. IdP provides a link to access all accounts in the organization. Key Components: **SCP (Service Control Policies):** Used to assign policies across accounts. **Organizational Units (OU):** Groups two or more accounts into a single entity. SCPs can be applied at the OU level.			
AWS Control Tower	**AWS Control Tower** Control Tower simplifies account management within an organization. When creating a Control Tower: A Security Directory is created by default, which includes: Log Account Audit Account Account Factory: Enables quick account creation. SCPs can be applied as guardrails. To log in to newly created accounts, SSO is required.			
Topics	Explanation			

Amazon VPC	**VCP** stands for virtual private cloud that used to create the virtual private network inside the network there can be multiple resources, when we need to expose the services to internet, we need to divide the network, the network get divided into multiple subnets based on our network, inside the CIDR range. **Public subnet** which contain public IPs for the services .**Private Subnets**, in that we use communication between the subnet to subnet we need to config the Route table. **Route table** have the route to specific IP range to subnets. **intergateway** is the entry point of vpc and public subnet,that used to allow the all the request and reponces to the internet **NAT gateway :**when the services or instance that present in the vpc need to acces the internet ,the natgateway get creted in public subnet and route the internet to spcific subnet or instance. **NACL(Network Access Control List)** act as firewall for the subnets, **Security group** is set of rule to allow and deny traffic; like firewall. It assigned to specific VPC or instances of services, **Elastic IP** is the services used to assign the public IP to services instance.			
VPC endpoint	The **VPC endpoint** that used to access the resources or services that not present inside the VPC or not include the component of VPC.that services that not have the compute and networking components. There are two types: **Gateway endpoint:** the endpoint provided by the services in order to access the particular services, it specifically only allow S3 and DynamoDB, it connect and resolve the endpoint internal AWS network, it is free of cost using this services **Interface:** the endpoint is created using ENI, that need to config in route table so only access to particular ENI. It make charge based on data transfer using interface, use at most 100+ services.			
Topics	Explanation			
VPC Peering	**VPC peering** is network connection between two VPC, to exchange the data and access the resources in one VPC to another. VPC peering only connect two different VPC in same account or different account, different availability zone, and other region. It use non-transit network connection. It establish the secure connection between the two VPC, and it is connecting through internally, the traffic flow within the network of AWS. To establish the connection create the feature of VPC, VPC peering, need to choose the name of peering, local VPC, choose the destination like another account, or region or same account. Then choose VPC or VPC ID for region, another account use, account ID with VPC. need to update the route table with vpx path to route another vpc			
AWS Client VPN	**VPN** is secure connection between device or network by establishing the secure tunnel, the data travel to the tunnel is not trackable and encrypted format, connection over the internet, one types of VPN is point to site. **AWS Client VPN** is used to connect the VPC to the device (point) using VPN tunnel and VPN client, it is the software need to installed in target device in order to communicate with VPC. To establish the connection need to create VPN client and download the config file to connect with VPC, need to upload the file in AWS client to make the communication between them.			

Site to Site with VPN	The Site-to-Site VPN is used to connect the two different network, cloud to on-prem (in secure connection use VPN tunnel, that can be establish the connection between them. Here we need to create the following component: **Customer Gateway:** use the public IP of the on-prem network, create the customer gateway, the customer gateway provide the configuration file, that need to configure manually in the on-prem network. **Virtual Private Gateway:** to monitor the data usage or the private connection of VPC need to travel through the VPG, it allow all the private traffic to VPC. After creating the VPG, need to attach with the VPC need to connect to Site-to-Site. **Site-to-Site VPN**: after creating above two step need make the Site-to-Site connection between the VPG to Customer Gateway, after choosing it, the VPN create the public IP to communicate to the Customer Gateway public IP in the on-prem. After the VPN connection is established, need to configure the route table.			
Transit Gateway	The **Transit Gateway** is used to make the connection between the two or more VPC, the Transit Gateway act as the hub and route traffic to respective VPC by its route table to the spok(hub and spoke model). Need to create the Transit Gateway with CIDR range, after creation need to attach the TGW to the VPC, after attaching it propagate the VPC route table.			
Topics	Explanation			
Elastic Cloud Compute	EC2 is elastic cloud compute, it is one of the compute services of the AWS, by using EC2 able to create virtual machine or EC2 instances within VPC in AWS account. based on instance time the CPU and memory will be allocated, the Amazon. for the OS and application server or application need to run we choose using Amazon machine image, next to setup there different types in instance **General purpose:** burstable, whenever the workload of computer gets continuously change. t2-t3, m5-balanced for consistent workload. **Compute optimized:** c5, high ratio of compute to memory, focus on CPU **Memory optimized:** high memory intensive application like database there two types r4, good for in memory database. x1e good for application running. **Accelerated computing:** the compute that used for high graphics processing and GPU, p series (p3) **Storage optimized:** the storage is for high IOPS (input-output per second) and low latency, h1-for HDD, i3 for SSD, d2 for highest disk ratio. to initial login to machine we need to specify the key pair with algorithms. based on instance type the volume gets created, need to setup the security group for the machine to access the EC2, initially it allow ssh, http, https. after these configuration launch the instances, the output give the private key, public key stored on the EC2. we can connect using ssh.			
EC2 instance pricing	**On-Demand Instances**:Pay for EC2 by the hour or second with no long-term commitment. **Spot Instances:**Buy unused EC2 capacity at up to 90% discount.but AWS can stop the instance anytime. **Reserved Instances** :Commit for 1 or 3 years for a big discount up to 70% compared to On-Demand. based on specific services and compute. **Savings Plans:**commit to a certain amount of compute usage for 1 or 3 years.			
lambda	**lambda** is the serverless services, that support function as a services, the OS and runtime, networking, CPU and storage are already defined, user only put the set of code to run the lambda, the lambda is start when any action get triggered. it not run every time, the instance or specify task only performed in lambda, like form submission, single button action. it capable of running max time is 15 mins, not highly interaction application not be done in lambda			
Topics	Explanation			

Lambda Trigger	The **Lambda**, after uploading the code, can create the endpoint URL to trigger the Lambda. The triggers can be configured into various services like CloudTrail, S3. When any action triggers, the Lambda gets started and runs. At a time, 1 request gets processed; if more requests come, it uses queue trigger. If you want to run multiple requests at a time, you need to use Concurrency. **Concurrency:** if enabled, based on requests, the function gets processed and increases the Fargate the request at a time. **Example:** When a user tries to upload the file from a device to S3, when the S3 gets the file to store, the Lambda gets triggered and gets the file, changes it into zip, and stores it to S3.			
Elastic Container Services	Elastic Container Services: based on the image, the container gets deployed as application services. In AWS, there are three main components to deploy ECS, **Cluster:** the cluster is a collection of containers; the cluster will provide the compute for each container in the cluster. **Task Definitions**: the task definition is used to create the container inside the cluster. **Namespaces**: the namespaces are used to create the isolated environment for the container; within the same namespace, containers can communicate. The EKS uses Fargate compute			
Elastic Kubernetes Services	**Elastic Kubernetes Services** provides the Kubernetes services; AWS takes care of the master node. While creating the cluster, you need to provide the necessary configurations below: Cluster name Need to give the VPC to create the cluster The cluster contains a node group created with instance type; the node group consists of max, min, desired node count.			
Batch	**Batch** is a collection of jobs; a single job is a process that takes compute to run. We can put jobs into a queue and then use a scheduler to run the jobs. While running the job, the compute gets utilized; the result gets stored in any storage area as defined.			
Amplify	**Amplify** is used to host the front-end and full-stack applications, which support HTML, CSS, JS, and mainly TypeScript. We can upload the code file or link the Git repository to process the code and deploy it as an application.			
Elastic Beanstalk	The **Beanstalk** is used to deploy the backend applications and full-stack applications. We can use the programming language for backend and support runtime with EC2 instance support; we can get into the instance and access it. It supports Python, Java, .NET, with runtime of each backend. It also manages database and networking while creating.			
Topics	Explanation			
Storage	Storage The storage is used to store and retrieve data from any disk or any network file system. There are three dedicated services for storage below: Simple Storage Services Elastic Block Store Elastic File System			

Simple Storage Service	The S3 is used to store the data directly into objects. The initially 5GB data stored in S3 is free tier. The S3 consists of buckets to store the data, and the bucket name is globally unique. In S3, we can host static HTML page hosting using a public URL. Here, no predefined size of storage is specified; it calculates the size of data that has occupied. It follows a pay-as-you-go model. It is a global service. while creating the S3, there are different features. One of the main features is S3 versioning. The versioning enables the various states of the same object after updation. It maintains separate states to manage the object. It is not overwritten, only if versioning is enabled. If any object that has versioning enabled is deleted, it is not permanently deleted; it is stored into a delete marker. After delete, we can retrieve it from the delete marker.			
Srorage class in s3	S3 Storage Classes After creating the S3, based on usage and storage retrieval performance, we can specify the storage class to maintain cost effectively and based on workloads: **S3 Standard** for frequently accessed data **S3 Intelligent-Tiering** to analyze the data accessing patterns and reduce the cost for unused data **S3 Express One Zone** for your most frequently accessed data **S3 Standard-Infrequent Access:**for less frequently accessed data **S3 One Zone-Infrequent Access** for less frequently accessed data in single zone **S3 Glacier Instant Retrieval** for archive data that needs immediate access **S3 Glacier Flexible Retrieval** for rarely accessed long-term data that does not require immediate access **S3 Glacier Deep Archive** for long-term archive and digital preservation with retrieval in hours at the lowest cost storage in the cloud These all configurations are done in lifecycle policy.			
Elastic block store	**Elastic Block Store (EBS)** EBS is the storage used to attach instances, and also supports multi-instances attach. There is actual block disk storage provided by multiple volume types like SSD (solid state drive) and HDD (hard disk drive). There are different types in EBS to choose the disk. For SSD, gp3 and gp2 are standard used for most of the instances. **SSD-**backed storage for transactional workloads **HDD-**backed storage for throughput-intensive workloads **Snapshot** The snapshot is a backup of data at a particular point. After the snapshot, we can restore it into any other EBS to replicate or restore the data from snapshot. The snapshot can be stored in any other storage devices in AWS infrastructure. It helps to overcome the failure of EBS.			
Volume Types	**IOPS and Throughput** IOPS: measures the number of read and write operations a storage device can perform per second. **General Purpose SSD** (gp2 and gp3): volumes are designed for a wide range of workloads, support minimum IOPS **Provisioned IOPS SSD** (io1 and io2): volume optimized high IOPS Throughput: measures the amount of data transferred from and to the storage device per second **Throughput Optimized HDD** (st1) for large volume workload **Cold HDD** (sc1) for infrequent access volume			

Elastic File System	**Elastic File System (EFS)** The file system is used to store the data without specifying any filesystem configuration. You need to mount it to any directory and use the EFS. The EFS is part of a VPC. The EFS is region-specific, not tied to an availability zone. The EFS provides automatic backup across the AZs. **Configuration includes:** **Lifecycle management:** set the time period to back the storage files into modes for cost optimization: archive, infrequent, standard **Performance of throughput:** Enhanced – different enhanced performance modes, Elastic – it scales up and down based on I/O, Provisioned – we limit the throughput of file by predefined MB limit Bursting-Provides throughput that scales with the amount of storage for workloads			
Database:RDS	**Database** The database is the place where we store and retrieve the data with the help of any database management system. There are different types of databases based on structure it stored: SQL (Structured Query Language): stores data in table format with respected rows and columns No-SQL (Not Only SQL): stores data in JSON format, key-value pair; it allows scaling of attributes In AWS, there are services that provide relational database services, known as RDS, that support SQL database management systems such as MySQL, PostgreSQL, SQL, and so on.			
RDS configuration	While creating the RDS, we can specify the following configurations: Database engine: PostgreSQL, MariaDB, Oracle, IBM Db2 **Deployment** options: Multi-AZ DB cluster deployment Multi-AZ DB instance deployment (2 instances) Single-AZ DB instance deployment **Credential** for the user/admin to access the database **Instance** configuration which contains CPU allocation, **storage** for the instances, and IOPS **Network** configuration with VPC and attach the subnet groups **Subnet groups** are a collection of subnets that users define and also availability zones. **RDS Proxy:** The proxy establishes the connection with the database. When a user tries to access the proxy, it does validation and gives the data based on the request. The background connection of database never stops; it is always in shared connections.			
Topics	Explanation			
Aurora	The Aurora is the relational database provided by AWS, that provides 5x faster than MySQL and 2x faster than PostgreSQL. It supports AWS serverless. It is a global database cluster, not specific to a single region.			
Dynamo DB	Dynamo database is used to store and retrieval the data in JSON format. It is NoSQL. It supports serverless. Its key components are below. **Tables:** the tables is a document that store the actual JSON data. **Items**: the items is data get presented in the snippet {}, inside the tables. **Attributes:** the attributes is type of key or datatype of items (column). **Primary key**: it is used to identify the items uniquely. After create, we need to access the services by URL.			

Elastic Cache	The Elastic Cache supports Redis engine. It is the in-memory data processing cache. It is used for **caching** the user frequently accessed data into the in-memory to make the instant retrieval. It is used to make the **session management** from server to client. It also **queueing** the request and process the request. **Rate limit** – limit the request to the particular server. **Example:** when it is a monolithic application need to communicate within the modules or function, not directly call the module. So here we queue the request to cache, and get the request from the cache and process it and generate output to the module.			
Security:Secrete manager	**Secret Manager** is the place where all the credential and confidential data get stored, like password, API key, and more. To access the credentials in Secret Manager, create the role to access the Secret Manager and attach into which application or services need to access the secrets. Example: When we are creating the backend application need to provide the RDS credential to backend. In that case we not hard code the credential because it is vulnerable. It is visible in any SCM or code level. To overcome this, store the credential with key and value in the secrets and create the role to access the Secret Manager and assign the role to particular EC2 that has backend application. The AWS SDK or client get communicate to Secret Manager and get credential while running use.			
Key Managemnt Services	The **KMS (Key Management Services)** is used to create, manage the cryptographic keys for encrypting and decrypting the data in AWS services. The key is used to encrypt and decrypt the data with different key. If we use single key for encryption and decryption it is **symmetric key**, and if used one key for encryption and another key for decryption, it is **asymmetric key** have two key. In AWS, there are two types of managing keys: **AWS managed keys**: the AWS managed keys is used to manage the services keys when it created, and also for accessing the services. It can manage by customer only managed by the AWS. **Customer managed keys**: the keys are we created to encrypt and decrypt the data that is self managed. We can create new and delete and manage. When the key is created it is used to encrypt the normal text into cipher text and then the key get encrypted and store into KMS as AWS managed keys.			
Web Application Firewall	**Web Application Firewall** is the dedicated firewall for the web application and the API of the application. The WAF monitor and check the incoming request IP, payload, parameter, to avoid cyber attacks like SQL injection. We can protect it by various action below: **Allow:** to allow the traffic directly to access the resources **Block:** to block the traffic **Challenge**: to validate the browser behavior by send the JavaScript then grant an access **Captcha:** to validate by human verification challenge then grant access **Count:** the get the request and log the incident for monitoring and analysis **Allow:** to allow the traffic to remaining NACL The actions are performed by predefined WACL (Web Access Control List). The WACL is attached to these services: CloudFront distributions, Application Load Balancers, API Gateway APIs. The WACL consists of rules. The rules is collection of statement. Each rule have separate action to protect it from the threats. Make the collection of rules to form the rule group. It also attach to WACL.			

Cloud Trail	The **CloudTrail** is event monitoring tool used to access audit log of the services in AWS. There are two types of event: **Management event:** the event that performed in any services listed here. **Data event:** the event about what are action made within the services like S3 write on which bucket. **Lake:** the lake is dump all the events recorded and used the query to get the data. It is useful to aggregate the data and display as user want. Whenever an account is created a default trail is created for the account. We can also create the trail as we need. Events get monitored by API calls and services logs.			
Route 53	**Route 53** is domain name service, that translate the domain name into IP. If the domain name is existing, we can migrate the domain from any other registry to AWS. The main component of Route 53 is **hosted zone.** It consists of public and private records. Inside the hosted zone we need to configure the subdomain for resources. If we have domain in another registry, we need to take the name server of Route 53 and configure to existing domain registry.			
Cloud Front	**CloudFront** is edge computing devices used to connect a front end data across all the region. The feature of CloudFront is **distribution:**It get the front data from the resources or IP or domain and create the distribution to deliver the front end data to all devices that in CloudFront when it access or make initial request. **Invalidation:** when any update made by the developer and it is deployed but user can't access the actual updated feature, in that case the edge compute have the old version of code. To delete that we need to perform invalidation.			

Topics	Explanation			
Load Balancer	**Load Balancer** distributes the incoming traffic across multiple compute resources, like EC2 or containers. To identify the healthy server, it uses a health check function to check every 5 seconds whether the server is healthy or in a dead state. If healthy, it routes the traffic based on algorithms like round-robin. The routed traffic is sent to resources in a target group, which is a group of resources attached to the LB for high availability and no downtime. There are different types of ELB based on OSI layers: **Application Load Balancer (ALB)** – Operates at the application layer (Layer 7) and supports HTTP and HTTPS. **Network Load Balancer (NLB)** – Operates at the transport layer (Layer 4) and supports TCP, TLS, UDP, and QUIC. **Gateway Load Balancer** – Operates at the network layer (Layer 3). **Classic Load Balancer** – Combination of application and network load balancer.			
Simple Queue Service	**SQS** is the process of queuing requests from one service to another to achieve sequential or FIFO order. When a huge number of tasks are sent at a time to a server, it may consume more compute and can also cause server crashes. To avoid this, we can use SQS. SQS puts all the requests into a queue and releases them one by one. If one request's destination is accepting the request, it waits and then sends the next request after the destination is alive.			
Simple Notification Service	**SNS** is used to send notifications to client endpoints, Lambda, or SQS. In AWS, you need to create **Topics** with two types: FIFO – Only supports SQS Standard – Supports all protocols For destinations, you need to create **Subscriptions** with endpoints. The endpoints can use protocols like HTTP, HTTPS, email, SMS, Lambda, or SQS.			

Database Migration Service	**DMS** is used to migrate databases from source to destination using endpoints and replication instances. During migration, compute resources are needed, so you create instances called replication instances. To migrate: 1.Create endpoints for source DB and destination DB with URL, port number, username, and password. 2.After creating endpoints, create a task with source and destination endpoints. 3.Compute can be provisioned or serverless, and instances can run in single-zone or multi-zone availability.			
Simple Email Service	**SES** provides an SMTP server to handle mail communication for your domain. You need to choose the domain or verify a domain already purchased. Based on that, you can create email IDs for each user and manage email communication. The cost is calculated based on emails sent and received.			
Topics	Explanation			
AWS Step Functions	The **Step Function** is used to orchestrate AWS services. We can create the logical flow of execution of services, and if an error occurs, we can have a separate logic flow for the error. When you trigger the task to run the Step Function, to run the flow we need compute. The **state machine** provides the execution for the tasks to run. The state machine has a pictorial representation of the workflow, which has start and end states. In between, services need to be executed.			
AWS Glue	AWS Glue is an ETL service. E – Extract the data from different sources. T – Transform is used to filter and modify the data as needed. L – Load stands for loading the data into destination databases or storage.			
CloudWatch	**CloudWatch** is used to monitor AWS services and applications inside AWS, and also to monitor metrics and logs. There are different types of logs, such as application logs and network logs. It provides information about CPU utilization and logs of particular services. There are two main functions in CloudWatch: **Alarm:** The alarm is used to trigger actions like notifications, Lambda, or Auto Scaling Groups based on predefined threshold values and conditions. **Log Group:** This is the directory where we can group log streams based on service types. Inside the log streams, the logs are presented.			
X-RAY	**X-Ray** is the service used to identify bottlenecks between services and to track requests and responses inside the application. By using this, we can identify where request time gets delayed and where errors occurred. We get detailed information about requests. The initial request that arrives at a service is called a segment, and all the requests that travel between services are called subsegments. Separate requests from one service to another have separate trace IDs to monitor services. Only specific AWS services support X-Ray. If other services need to use X-Ray, we can use the AWS client SDK to integrate X-Ray.			
DataSync	**DataSync** is used to copy files and objects from one storage service to another storage service in AWS or on-premises using the DataSync agent. It supports live copying of objects at predefined intervals and also one-time copy, not continuous sync.			
Cognito	**Cognito** is used to provide user authentication and management services for applications. It supports user data management such as username and password modification. There are two main components in AWS Cognito: **Cognito User Pools:** Used to create users, manage users, and authenticate users to applications. It creates a directory for the application where user information is stored. After sign-in, it redirects to the respective application endpoint. **Cognito Identity Pools:** It creates identities to access AWS resources like S3 and Lambda. It provides temporary access to users with a set of credentials that can be imported from any identity provider, including Cognito User Pools.			

Event Bridge	**EventBridge** is used to trigger events based on actions performed in AWS services. When an event is triggered, it runs the target service. It acts as a bridge between two events. Events are specified in **rules**, and rules are grouped into an **event bus**. By default, it uses the default event bus. It also supports JSON and a drag-and-drop model. It consists of a **scheduler** to schedule events using cron. **Pipes** define the flow of how data is extracted from a source, filtered based on conditions, enriched by modifying the data, and finally sent to the target service where the event is triggered.			
API Gateway	**API Gateway** is used to authenticate and authorize APIs using various authentication methods like API keys or Cognito users. The gateway has two parts: integration, which defines where the endpoint routes to, and custom domain endpoints. Based on authorization, access to endpoints is granted.			
Software Development lifecycle	**SDLC** describes the various phases of software development—how software is created, tested, deployed, and monitored. The stages of SDLC are: **Plan:** Identify the project goals or problems. **Requirements:** Collect necessary data and requirements from users and stakeholders. **Design:** Architect the solution and build a demo or prototype based on requirements. **Develop:** Implement the design and develop the software. **Testing:** Test every function and part of the software and fix bugs. **Deploy:** Deliver the software into production and make it visible to end users. **Monitor:** Monitor performance, collect feedback, and analyze it. This is also known as the Agile model because the monitoring phase iterates back to the planning phase, making it a recursive process. In the Waterfall model, the process is not iterative. Once a phase is completed, it cannot be revisited, and replanning is not part of the waterfall model.			
Scrum Process	The **Scrum** process involves three entities: **Scrum Master**, **Product Owner**, and **Scrum Team**. The Scrum Master creates the project plan and estimates using sprints. A **sprint** is a fixed range of days to complete assigned tasks. If the project is not completed within a sprint, it moves to the next sprint and is labeled as a product backlog. If a sprint needs to be extended due to insufficient time or other reasons, it creates a sprint backlog. Normally, a sprint takes 1 to 4 weeks. The Product Owner or Scrum Master conducts daily meetings with the team to update the status—what is done, what is planned for today. After each sprint, work is reviewed and retrospectives are conducted. After every sprint, the product may be deployed or the work completed.			
DevOps	**DevOps** is the collaboration between development and operations to deliver software to production and ensure automated deployment. The DevOps lifecycle phases include: **Plan:** Requirement gathering and designing prototypes. Tool: Jira **Code:** Implement the plan and develop the code. Tools: GitLab, GitHub **Build:** Package the code with required dependencies. Tools: Gradle, Sonatype Nexus **Test:** Validate features and functionality. Tool: Selenium **Release:** Prepare code releases and versions. Tool: Jenkins **Deploy:** Deploy the latest release to production. Tools: AWS, Azure, Docker **Operate:** Fix bugs, handle errors, and scale resources. Tools: Chef, Ansible **Monitor:** Monitor logs and feedback and reinitiate planning if required. Tool: Grafana			

Branching strategy	**Branching** is the process of storing code in different directories. The main directory is named main or master based on the SCM tool. Different branching strategies include: **Dev-Master:** Two branches—dev for development and master for production. **Trunk-Based:**Initaily one branch created ,for any update, feature branches are created for new features and merged back into the main branch. Feature branches are short-lived, while the trunk is long-lived. **Environment Branching Strategy:** Separate branches for different environments such as dev, test, and prod. Changes are reflected across environments. **Release Branching Strategy:** Separate branches for environments and individual releases like dev, QA, release-1, release-2. **Hotfix:** When a bug needs to be fixed in production, a hotfix branch is created, the issue is resolved, and the fix is merged back into production and other branches.			
Topics	Explanation			
Git	**Git** is an open-source version control system that uses a distributed system. Key components of Git: **Repository**: Used to store the code folder. **Branch:** Different folder of code inside the repository derived from the master or main branch. **Commit:** Used to store updates and snapshots of changes; it achieves versioning.			
Git Commands	**git init**: Used to initially create a new Git repository in the current directory. **git add <file_name>:** Adds the file to the local repository. If you need to add all files, use dot in place of the filename. **git commit -m "Custom message":** Used to save the change into the Git local repository. After commit, it generates a unique ID for each commit and also mentions a message to identify the purpose of the commit. **git clone "url_of_repository":** To replicate the GitHub repository code into the local repository. **git pull:** To download the changes or updated code and apply the changes to the current directory. **git fetch:** Retrieves changes from a remote repository, including new branches and commits. **git checkout -b "branch_name":** Used to create a new branch. **git checkout "branch_name":** Used to change the branch. **git log**: Lists the commit history in the current branch. **Merge conflict:** When you create a new branch from the main branch to update a feature, and someone makes changes in the main branch, when the new branch tries to merge with the main branch, this creates a merge conflict. **git merge <branch_name>:** Used to merge the specified branch into the current branch. The commit history is shown in a separate section, not in sequence. **git cherry-pick <commit_hash>:** Used to merge a specific commit from one branch to another branch. If a conflict occurs, it lists the changes in the code. Manually fix them, then apply the below commands: **git add <file_name>** **git cherry-pick --continue:** This automatically creates the commit. **git rebase <branch_name>:** Rebase merges the code and makes the commit history			
Source Code Management Tool	**Source Code Management Tool:** This tool is used to store the source code of software or projects. It provides managed versioning and runs pipelines for automated deployment processes. To run pipeline scripts, it requires compute resources called runner compute. Tools: 1. Git 2. GitHub 3. GitLab 4. Bitbucket 5. Azure DevOps / TFS			

Github	**GitHub** GitHub is an open-source platform that provides Git repository management and hosting, static website hosting, and collaboration features. It is widely used by developers to manage versions of code. Features: **Remote Repository:** Supports public and private repositories. **Pull Request:** Used to review code and merge one branch into another. **Issues:** Create tasks or issues, assign users, and set milestones. **Actions:** Automates CI/CD workflows. **Collaboration & Access Levels:** **Read:** View and clone repository; open issues. **Triage:** Manage issues and pull requests without modifying code. **Write:** Push code, create branches, and merge pull requests. **Maintain:** Manage repo settings, labels, and milestones; no access to sensitive data. **Admin:** Full control access			
Gitlab	**GitLab** GitLab is a web-based DevOps platform that provides Git repository management along with a complete CI/CD pipeline. It can be self-hosted or used as a cloud-based service. **Key Features:** **Integrated CI/CD**: Built-in continuous integration and continuous deployment pipelines. **Merge Requests:** Similar to pull requests in GitHub, enabling code review and team collaboration. **Issue Tracking**: Comprehensive issue tracking and project management features. **Container Registry**: Built-in Docker container registry for storing and managing container images. **Repository Grouping:** Allows multiple repositories to be grouped under a single group or namespace for centralized management, access control, and visibility.			
Azure DevOps	**Azure DevOps** Azure DevOps is a suite of development tools from Microsoft that includes version control, project management, build automation, and release management. **Key Features:** **Pull Request:** Used to review code and merge one branch into another. **Version Control:** Supports both Git repositories and Team Foundation Version Control (TFVC). **Boards:** Agile project management tools for tracking work items, sprints, and backlogs. **Pipelines:** CI/CD capabilities for automating builds and deployments. **Test Plans:** Tools for managing test cases and executing tests. **Artifacts:** Package management for sharing code and dependencies.			
Bitbucket and Team Foundation Server	**Bitbucket:** Bitbucket platform that provides Git repository hosting along with features for code collaboration and CI/CD. It is part of the Atlassian suite of tools. Integration with Jira: integration with Jira for issue tracking and project management. **TFS /Azure DevOps Server:** TFS (now known as Azure DevOps Server) is a suite of development tools from Microsoft that provides version control, project management, build automation, and release management in a centralized on-premises environment.			