Task-1:

**What is SDLC?**

SDLC stands for Software Development Life Cycle,it is a structured process used by development teams to design, build, and maintain software. It's a systematic approach that breaks down software development into distinct stages to ensure high-quality software is produced efficiently.

Task- 2

**Why is SDLC ?**

The Software Development Life Cycle (SDLC) is important because it provides a structured, systematic approach to software development, leading to higher quality, more efficient, and cost-effective projects. The Software Development Life Cycle (SDLC) is crucial for structured software development, offering benefits like improved planning, quality control, and risk mitigation. While it's possible to create software without a formal SDLC, especially for smaller projects or individual development, following an SDLC ensures better project management, consistency, and a higher likelihood of meeting project requirements and business goals.

Task -3

**Stages OF SDLC are as follows:**

* 1. Planning:  
  This initial phase involves defining the project's goals, scope, and feasibility, including resource estimation and cost-benefit analysis.
* 2. Analysis:  
  In this stage, the team gathers and analyzes user requirements to understand the needs of the final product.
* 3. Design:  
  The team translates the requirements into a detailed blueprint, including the software architecture, data structures, and user interface.
* 4. Development:  
  This is the coding phase where the software is built according to the design specifications.
* 5. Testing:  
  The software is thoroughly tested to identify and fix bugs, ensuring it meets the requirements and performs as intended.
* 6. Deployment:  
  The completed software is released to the end-users, making it available for use.

7. Maintenance:  
 After deployment, the software is maintained, including bug fixes, updates, and enhancements to ensure it continues to meet the evolving needs of the users.

Task 4:

Stages of SDLC are as follows:

1. Planning and requirement: In this initial stage decide what needs to be done. It’s like making a plan for a trip: where are we going? How will we get there? How long will it take?
2. Analysis: In this stage the teams gathers and analysis user to understand the needs of final requirement
3. Design: create a blue print for how the software will work. This is like making a map for your trip, showing the route and places to visit.
4. Development: In this stage we can actually built the software by writing the code. This is like packing your bags and getting everything ready for the trip.
5. Testing: We can check if everything works correctly. It’s like double-checking your luggage and plans before leaving for the trip to make sure you didn’t forget anything.
6. Deployment: We can release the software so people can start using it. It’s like finally going on your trip and enjoying the destination.
7. Maintenance: After it's out there, you keep an eye on it, fix any problems, and add new features if needed. It’s like making sure your trip goes smoothly and adjusting plans if something goes wrong.

TASK - 5:

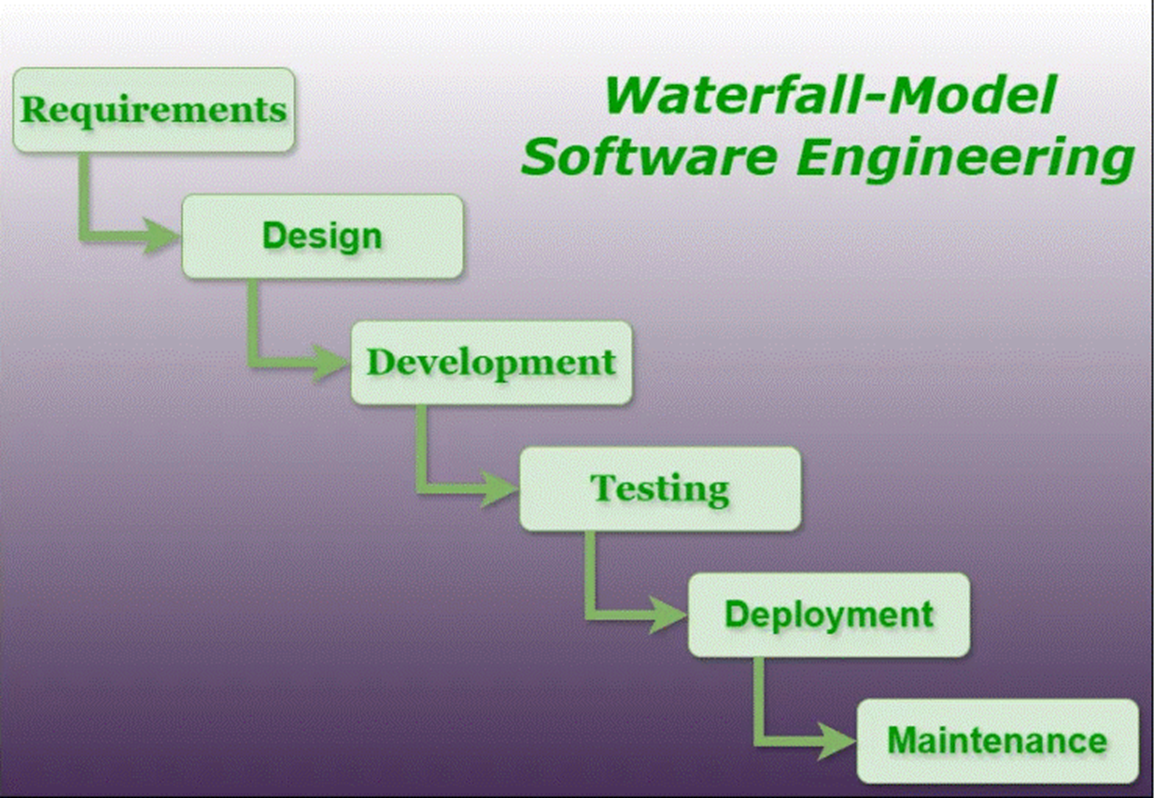
**SDLC Models are as follows:**

### Water-fall Model:

The Waterfall model is one of the earliest SDLC models. It follows a linear, sequential approach where each phase must be completed before moving on to the next.

Phases:  
Requirement Analysis  
System Design  
Implementation (Coding)  
Integration and Testing

Deployment  
Maintenance



* Advantages: Easy to understand and use; clear documentation.
* Disadvantages: Rigid structure; difficult to make changes once the process starts.

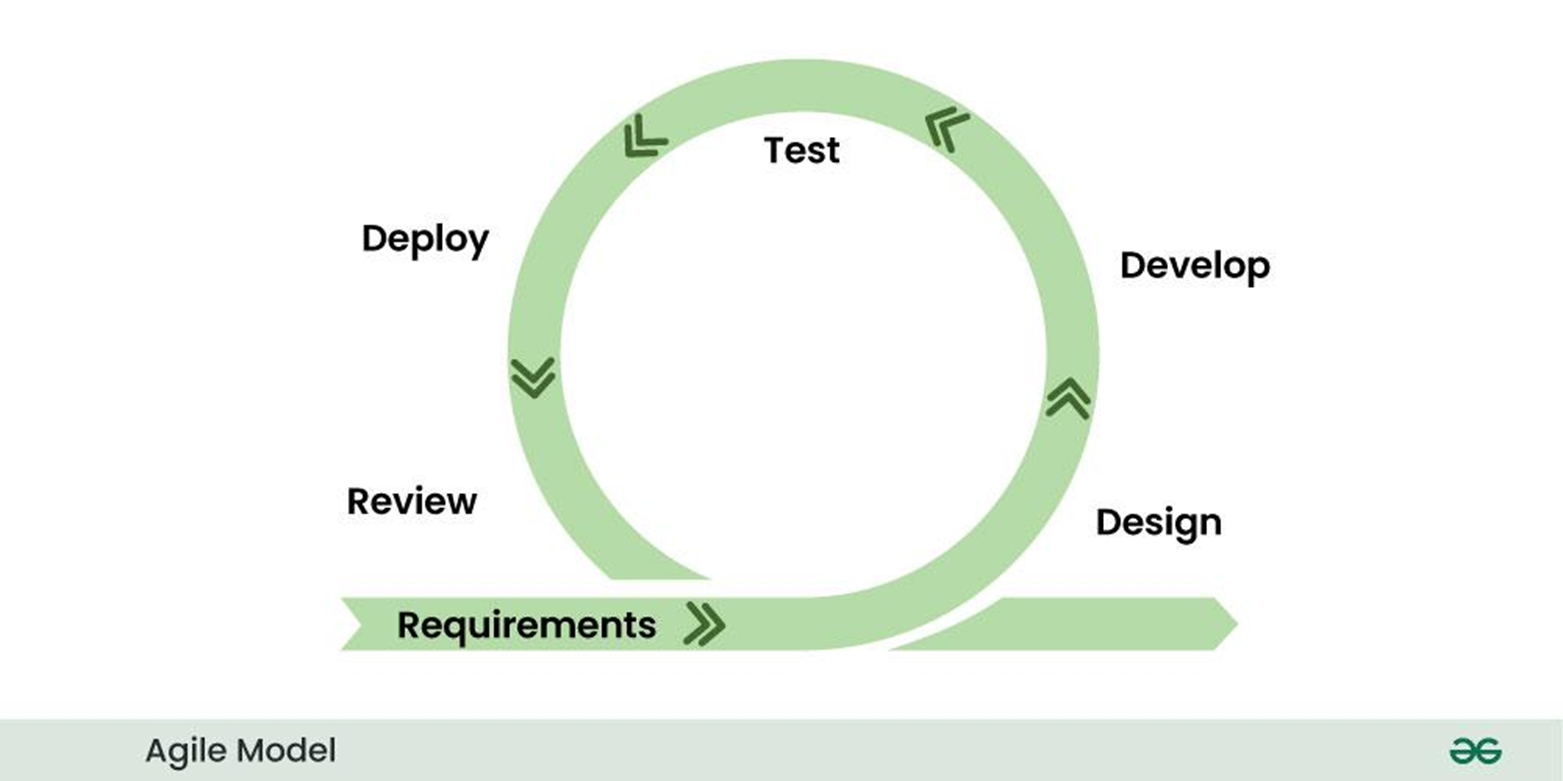
Applications:

* Projects with clearly defined requirements and no expected changes.
* Government or military software projects with rigid standards.
* Small or medium-scale projects with well-understood technology.

### 2. V-Model (Verification and Validation):

The V-Model is an extension of the Waterfall model. It emphasizes testing and validation activities corresponding to each development phase.

Phases:  
Requirements Analysis  
System Design  
Implementation  
Testing (Unit, Integration, System, Acceptance)  
Deployment  
Maintenance



* Advantage: Early detection of defects; systematic testing approach.
* Disadvantages: Inflexible, like Waterfall; changes are difficult once development starts.

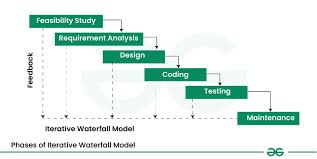
Applications:

* Projects where high reliability and testing at each stage are crucial.
* Systems requiring formal documentation and strict testing, like embedded or aerospace software.

### 3. Iterative Model

The Iterative model focuses on repeating (iterating) over the development cycle, where each iteration produces a part of the product. It gradually refines and improves the product in each cycle.

Phases:  
Requirement gathering  
Design  
Implementation  
Testing  
Evaluation  
Planning for the next iteration

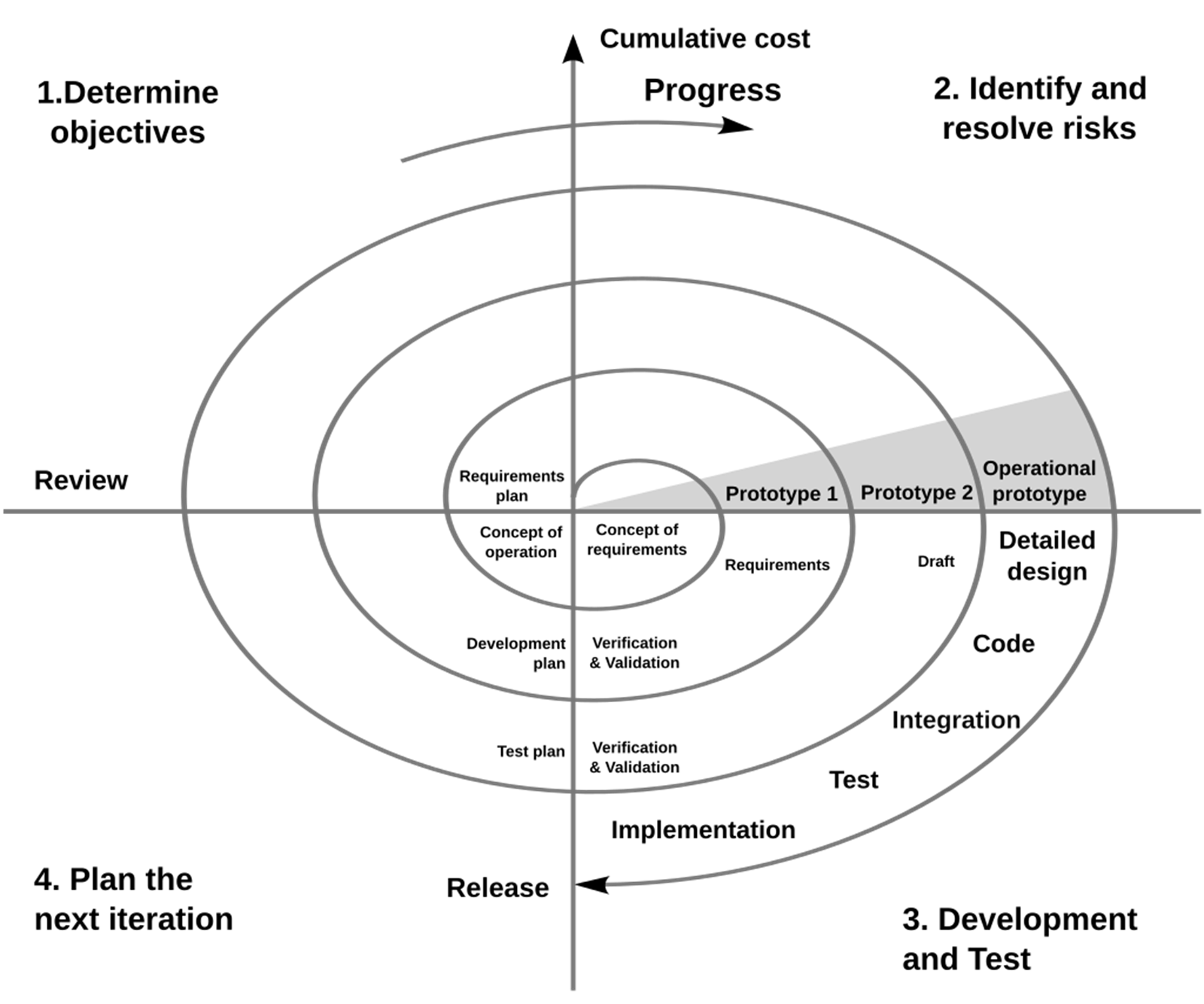


* Advantages: Flexibility to change requirements; regular feedback loops.
* Disadvantages: Can lead to scope creep; may require additional resources.

### 4. Spiral Model

The Spiral model combines iterative development with a risk-driven approach. It focuses on identifying and addressing risks at every phase.

Phases:  
Planning  
Risk Analysis  
Engineering (Development & Testing)  
Evaluation and review



* Advantages: Good for large, complex projects with high risk; flexible to changes.
* Disadvantages: Complex to manage; requires experienced project management.

Applications:

* Large, complex, and high-risk projects.
* Projects with evolving requirements or where regular risk assessment is necessary.
* When early prototypes help refine user requirements.

### 5. Agile Model

The Agile model emphasizes iterative development, customer collaboration, and flexibility. It focuses on delivering small, incremental changes in short time frames (called sprints).

* Phases (in each sprint):  
  Requirements Definition  
  Design  
  Development  
  Testing  
  Deployment  
  Review and Feedback
* 
* Advantages: High flexibility and adaptability; fast delivery of working features.
* Disadvantages: Lack of predictability; can be hard to scale for large projects.

Applications:

* Projects requiring frequent updates, user involvement, and rapid delivery.
* Ideal for dynamic environments with changing requirements.
* Startups and product-based companies needing MVPs and fast iterations.

TASK-7:

**What is Scrum?**

Scrum is a simple way for teams to work together to build things step by step in software development. In other words Scrum is a method that helps teams work on a project in small parts, called sprints, so they can finish faster, fix problems quickly, and improve as they go.

Scrum is a framework used to implement Agile.

Think of Agile as the *mindset* or *approach* to building software quickly and flexibly, and Scrum as one specific way to follow that approach[. In](http://approach.in) other words Scrum is a way to apply Agile by breaking work into short cycles, delivering usable software often, and improving with each cycle. Scrum is an Agile methodology used to manage software development projects in iterative cycles called Sprints. It fits into the Development phase of the SDLC but touches others (like Planning, Testing, Maintenance) due to its iterative nature.

**What is sprint?**

A Sprint is a short, fixed period of time (usually 1 to 4 weeks) during which a Scrum team works to complete a specific set of tasks from the product backlog and deliver a working part of the product. It is a core part of the Agile and Scrum methodology within SDLC. A Sprint is a mini project inside the bigger software project. It helps break down big tasks into smaller, manageable chunks so the team can build the product step by step.

**What are the do’s and don’ts of Sprint?**

### 1. Time-Boxing

* Stick to the Sprint duration (e.g., 2 weeks). Don’t extend it if work isn’t finished.
* This creates rhythm, predictability, and urgency.

### 2. Sprint Goal

* Every Sprint must have a clear goal. It aligns the team’s efforts.

### 3. No Scope Changes Mid-Sprint

* Once a Sprint starts, don’t add or remove tasks unless it's critical.
* New requests go to the Product Backlog for future Sprints.

### 4. Daily Scrum (Stand-up)

* Hold a 15-minute daily meeting for team members to share progress, blockers, and plans.

### 5. Definition of Done (DoD)

* Have a clear checklist that defines when a task is truly "done" (e.g., coded, tested, documented).

### 6. Sprint Review & Retrospective

* Always conduct these at the end of each Sprint to inspect the work and improve the process.

## Do's of a Sprint

| Do | 💬 Description |
| --- | --- |
| Set a Clear Sprint Goal | Define what the team aims to achieve during the Sprint. |
| Time-box the Sprint | Keep the Sprint duration fixed (commonly 1–4 weeks). |
| Have a Ready Backlog | Ensure the Sprint backlog is well-defined and prioritized before starting. |
| Hold Daily Stand-ups | Have a short (15-min) daily meeting to sync up. |
| Encourage Team Collaboration | Foster communication and shared ownership among team members. |
| Follow the Definition of Done (DoD) | Ensure every task meets the agreed quality standards. |
| Demo in Sprint Review | Show completed work to stakeholders at the end of the Sprint. |
| Run a Sprint Retrospective | Reflect on what went well and what can be improved. |
| Adapt and Improve Continuously | Use feedback to make the next Sprint better. |

## Don'ts of a Sprint

| Don’t | 💬 Description |
| --- | --- |
| Change Scope Mid-Sprint | Avoid adding or removing backlog items once the Sprint begins. |
| Skip Planning, Review, or Retrospective | These ceremonies are essential—don’t treat them as optional. |
| Overload the Sprint | Don’t commit to more work than the team can handle. |
| Ignore Blockers | Never leave issues unresolved—raise and address them quickly. |

**What is backlog?**

1.A backlog is a prioritized list of work that needs to be done for a product or project. It helps organize what to build, in what order, and when.

2. In SDLC terms, it acts as a dynamic to-do list that evolves as the product evolves.

**What is stories?**

1.A user story is a short, simple description of a feature told from the perspective of the user.

2.A User Story is a short, simple description of a feature or need from the end-user’s perspective. It's the most common item in a backlog.

Example:

As a bank customer, I want to receive an email when a transaction is made so that I can track my account activity.

**What is Scrum aircrafts, Product Backlog, Sprint Backlog, Burn-Down Chart and Increment?**

Scrum Artifacts are key information tools used by the team to track and communicate work.

They help ensure transparency, progress visibility, and alignment.

## 1. Product Backlog

* What it is: A prioritized list of everything the product might need — features, enhancements, bugs, technical tasks, etc.
* Owned by: The Product Owner
* Used for: Planning future Sprints and releases
* Dynamic? Yes — it evolves constantly as the product and market change.

Example Item:

As a user, I want to filter search results so I can find what I need faster.

## 2. Sprint Backlog

* What it is: A subset of the Product Backlog selected for a Sprint, plus a plan for how to deliver it.
* Owned by: The Development Team
* Used for: Guiding day-to-day work during the Sprint
* Includes:  
  + Selected User Stories
  + Tasks and technical subtasks
  + Sprint Goal

Example:

From the Product Backlog, 5 user stories are chosen for this Sprint. The team breaks each one down into tasks (e.g., design UI, write backend logic, test).

## 3. Burndown Chart (optional but commonly used)

* What it is: A visual chart showing how much work remains in the Sprint over time.
* Used for: Tracking progress daily and seeing if the team is on track.
* Axes:  
  + X-axis: Time (days of the Sprint)
  + Y-axis: Remaining work (story points or hours)

Benefit:

If the team is "burning down" tasks slower than expected, it signals a potential delay.

## 4. Increment

* What it is: The sum of all completed Product Backlog items during a Sprint.
* It must be: Done, usable, and potentially shippable.
* Inspected: During the Sprint Review
* Carries forward: As the base for the next Sprint's work.

Example:

If Sprint 3 delivered a login system, user dashboard, and notification center — all tested and working — that’s the Increment.

What are ports and protocols?

### Port = Communication Channel on a Device

A port is a numbered "door" or endpoint on a device used by protocols to send or receive data.

* Think of your computer as a building, and ports are the doors.
* Each protocol/service uses a specific port number.

### Protocol = Communication Rulebook

A protocol is a set of rules that define how data is transmitted between devices on a network.

**What are the types of Networks?**

### 1. PAN (Personal Area Network)

* Scope: Very small (within a few meters)
* Used by: One person
* Devices: Smartphones, laptops, headphones, smartwatches
* Example: Bluetooth connection between your phone and wireless earbuds

### 2. LAN (Local Area Network)

* Scope: Small area (like an office or home)
* Used by: A group of computers in a single location
* Devices: Computers, printers, routers
* Example: Wi-Fi network in a home or office

### 3. CAN (Campus Area Network)

* Scope: Multiple buildings in one location (e.g., university, business campus)
* Used by: Institutions or companies with multiple departments
* Example: University network connecting multiple departments and libraries

### 4. MAN (Metropolitan Area Network)

* Scope: Covers a city or large campus
* Used by: Government or large businesses
* Example: City-wide police/fire department network, or a city Wi-Fi system

### 5. WAN (Wide Area Network)

* Scope: Very large area (across cities or countries)
* Used by: Governments, ISPs, corporations
* Example: The Internet is the biggest WAN

### 6. Internet

* Scope: Global
* Used by: Everyone
* Description: A global network of networks that connects billions of devices worldwide

### 7. VPN (Virtual Private Network)

* Type: Not a physical network, but a secure connection over another network (usually the Internet)
* Used for: Privacy, remote work, secure data access

What are the types of servers ?

A server is a powerful computer that provides resources, data, services, or programs to other computers, known as clients, over a network.

Types of servers are as follows:

| Server Type | What It Does | Example |
| --- | --- | --- |
| Web Server | Hosts websites and delivers web pages to users via HTTP/HTTPS | Apache, Nginx, Microsoft IIS |
| File Server | Stores and manages files so clients can access and share them | Windows File Server, Samba |
| Database Server | Stores, retrieves, and manages data in databases | MySQL, PostgreSQL, Oracle DB |
| Application Server | Runs specific applications and business logic for clients | Tomcat, WebLogic, JBoss |
| Mail Server | Sends, receives, and stores emails | Microsoft Exchange, Postfix |
| DNS Server | Resolves domain names (like google.com) to IP addresses | BIND, Cloudflare DNS |
| FTP Server | Transfers files over the Internet using FTP protocol | FileZilla Server, vsftpd |
| Proxy Server | Acts as a gateway between client and the internet, often for security or caching | Squid, Nginx as reverse proxy |
| DHCP Server | Automatically assigns IP addresses to devices on a network | Windows DHCP Server |
| Print Server | Manages printers and print jobs on a network | CUPS (Linux), Windows Print Server |
| Game Server | Hosts multiplayer online games, managing game sessions and player data | Minecraft Server, Steam Server |
| Virtual Server | Simulates multiple servers on a single physical machine | VMware, Hyper-V, KVM |
| Cloud Server | Hosted on cloud platforms and used for scalable computing resources | AWS EC2, Azure VM, Google Cloud VM |
| Media Server | Streams audio and video files to devices on a network.  What do you know about DNS? Domain Name Service.  DNS is like the phonebook of the internet. It translates domain names (like www.google.com) into IP addresses (like 142.250.190.4) that computers use to identify each other on the network.  🔍 You type a website name → DNS tells your browser where to find it.  Humans remember names (like openai.com), but computers use IP addresses.  DNS allows you to visit websites using names instead of typing complex numbers.  TASK-16  What are the different types of Network topologies? Types of Network Topologies1. Bus Topology  * Description: All devices are connected to a single central cable (the bus). * Pros:    + Easy to set up   + Cost-effective for small networks * Cons:    + If the main cable fails, the whole network goes down   + Limited scalability   Example: Small office networks in the past 2. Star Topology  * Description: All devices are connected to a central hub or switch. * Pros:    + Easy to manage and troubleshoot   + One device failure doesn’t affect the rest * Cons:    + If the central hub fails, the whole network goes down   Example: Most modern LANs (Wi-Fi routers use this) 3. Ring Topology  * Description: Devices are connected in a circular loop. Data travels in one direction (or two, in dual ring). * Pros:    + Equal access for all nodes * Cons:    + A failure in one device or cable can affect the whole network   + Slower than star topology   Example: Token Ring networks (mostly outdated) 4. Mesh Topology  * Description: Each device is connected to every other device. * Pros:    + Very reliable and fault-tolerant   + Data can take multiple paths * Cons:    + Expensive and complex to set up   + High cabling cost   Example: Military or mission-critical networks 5. Tree Topology (Hierarchical)  * Description: A mix of star topologies arranged in a hierarchy (like branches of a tree). * Pros:    + Scalable and structured * Cons:    + If the backbone (root) fails, entire sections can be lost   Example: Large organizations with departmental segmentation 6. Hybrid Topology  * Description: Combines two or more topologies (e.g., star + mesh). * Pros:    + Flexible and customizable * Cons:    + Can be complex and costly to manage   Example: Real-world enterprise networks.  **What is OSI Model ? Describe the 7 layers with description.**  The OSI Model (Open Systems Interconnection Model) is a conceptual framework used to understand and design how data moves from one computer to another in a network.  It breaks the communication process into 7 layers, each with a specific function.  The 7 Layers of the OSI Model (Top to Bottom) are as follows: 1. Application Layer  * It is Closest to the end user * It Interfaces with software (browsers, email clients) * Example Protocols: HTTP, FTP, SMTP  2. Presentation Layer  * It Formats data for the application layer * It Handles encryption, decryption, compression * Example: SSL/TLS, file formats like MP4, JPEG  3. Session Layer  * It Manages sessions (start, maintain, end) * It is Useful in remote logins, video conferencing * Example: RPC, NetBIOS  4. Transport Layer  * It Ensures reliable delivery of data * It Controls flow and error handling * Protocols: TCP (reliable), UDP (faster, no guarantee)  5. Network Layer  * It Handles routing and logical addressing (IP) * It Decides the best path for data * Protocols: IP, ICMP, IPsec  6. Data Link Layer  * It Handles MAC addressing, error detection, and frame delivery * It Splits data into frames * Protocols: Ethernet, PPP, ARP  7. Physical Layer  * It Deals with hardware, signals, and bit transmission * It Includes cables, switches, electrical signals * It Transmits raw binary data (0s and 1s) | Plex, Emby, Jellyfin |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |