**Day 1 (16 May 2025)**

**112552596 - vsisukhw - Sukhwinder Singh**

**TASK 1**

**SDLC**

It's a just a process that helps to build software efficiently.

From starting phase (means planning, analysing) to last phase (i.e testing, deployment, and maintenance), SDLC helps.

Its like interface that manages software projects and ensure software meets customer needs.

**TASK 2**

**Why we Use SDLC**

Because of its

1. Structured approach

2. Improved quality

3. Increased productivity

4. Reduced cost

**TASK 3**

**Different stages of SDLC**

There are **6 stages** of SDLC.

1. **Planning and Requirement** - This is initial phase which defines the project aim, creating a plan for it and understand what the software needs to do.

2. **Design** - this includes designing the software architecture like data structure and algorithm

3. **Coding** - writing code based on needs

4. **Testing** - this includes testing the software to ensure it meets the requirements, also it identifying bugs

5. **Deployment** - launch of software

6. **Maintenance** - monitoring the software regularly and fixing bugs or issues regularly, if any

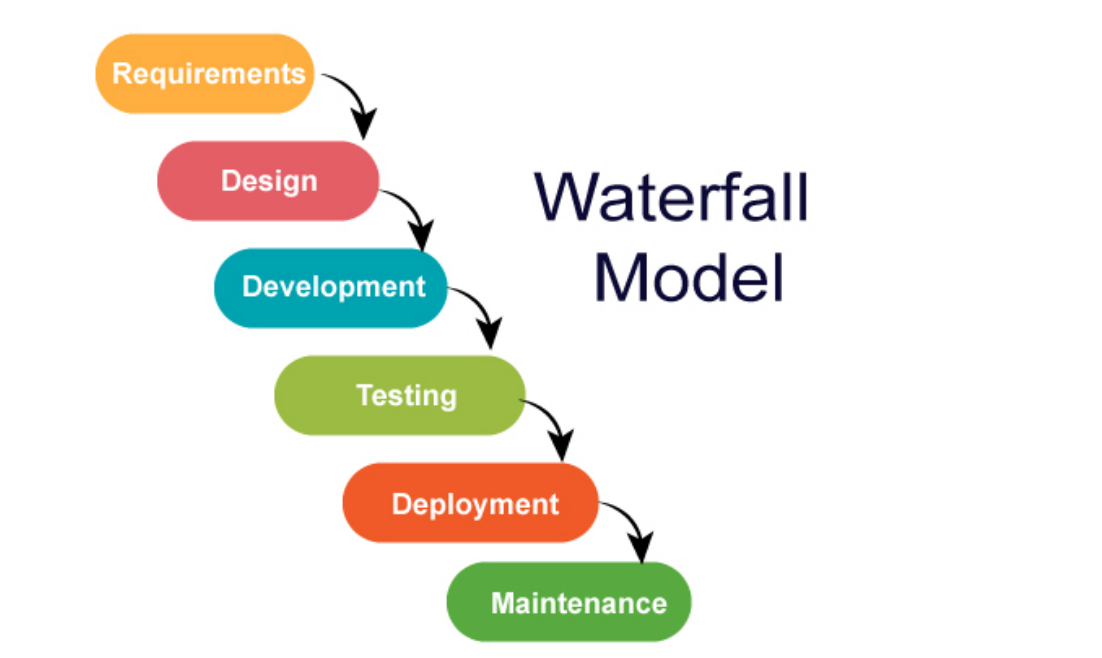
**TASK 4 , 5 and 6**

**Models of SDLC**

**(Definition, Its Image, Its Application, Its Advantages and disadvantages)**

There are **5 models** of SDLC :

1. **Waterfall Model** : it is a sequential model and it follows a strict linear approach and each phase (requirements, design, implementation, testing, deployment) must be completed before the next.



**Application**: Best for well-defined projects with fixed requirements like Government projects (e.g., defense, NASA), Banking systems.

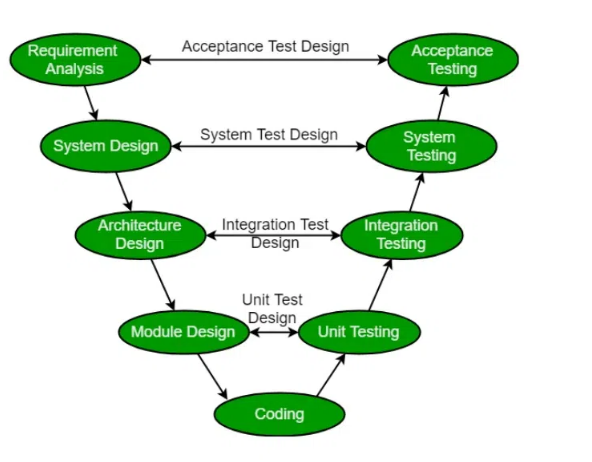
**Advantages**:

* Simple and easy to manage.
* Each phase has clear deliverables.
* Well-suited for structured documentation.

**Disadvantages**:

* Not flexible; changes are difficult to accommodate once the project starts.
* Poor model for long and ongoing projects.
* Late discovery of issues due to late testing phase.

1. **V- Model** : it is an extension of the waterfall model, in this model, while development it emphasizes on testing along side with each phase



**Application**: Suitable for projects where high reliability is essential, like healthcare or aerospace systems or automotive industries

**Advantages**:

* Emphasizes verification and validation early in development.
* Errors are caught early, reducing costs.

**Disadvantages**:

* Rigid, like the Waterfall model.
* Not ideal for evolving or unclear requirements.

1. **Agile Model** : it is a iterative model, it provides team with flexibility as it is allow teams to work on requirement as it arises and it is done in short cycles



**Application**: Ideal for dynamic projects with frequently changing requirements (e.g., web or mobile apps) like Spotify, Salesforce,etc

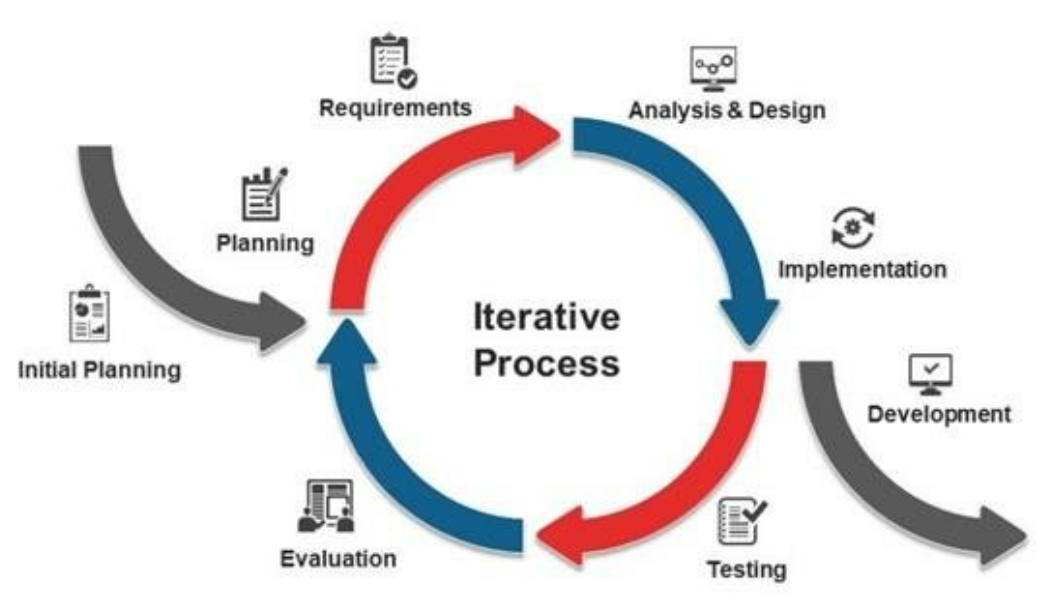
**Advantages**:

* Highly flexible and adaptive to change.
* Frequent feedback from users.
* Faster delivery of functional software.

**Disadvantages**:

* Requires frequent interaction with customers.
* Can lead to scope creep if not managed well.
* Less documentation.

4. **Iterative Model**: Divides the project into smaller iterations, where each iteration produces a working version of the software.



**Application**: Useful when requirements are not well-understood at the beginning. Like IBM, Microsoft

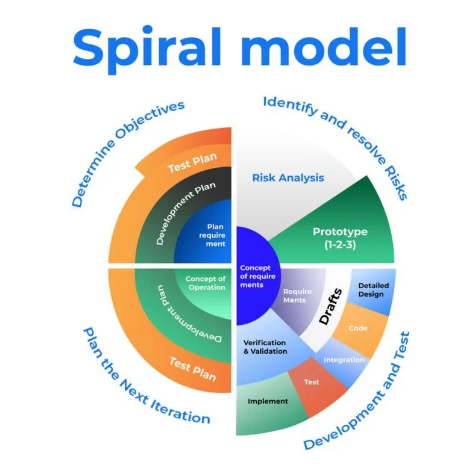
**Advantages**:

* Early working versions of software are available.
* Easier to test and debug during smaller iterations.

**Disadvantages**:

* May require more resources than Waterfall.
* Risk of incomplete or delayed iterations.

5. **Spiral Model** : it is also a iterative model but it is for high risk management , it is used where risk management is higher, it is done in cycles of development and testing



**Application**: Best for large, complex, high-risk projects. Like Defense and aerospace systems, missile control systems

**Advantages**:

* Emphasizes risk analysis.
* Allows for incremental releases and refinement.

**Disadvantages**:

* Expensive and time-consuming.
* Requires expertise in risk assessment.

**TASK 7**

**What is Scrum?**

Scrum is an Agile framework used to develop, deliver, and sustain complex products, especially software. It is one of the most popular methods for managing software development in Agile environments.

In simple words,

Scrum is a way of working in a team to build software (or any product) step by step, in short time periods called **sprints** (usually 1 to 2 weeks). The goal is to finish small, useful parts of the product quickly, get feedback, and keep improving.

**TASK 8**

**What is Sprint?**

Sprint is a short time period during which a team works to complete a set amount of work.

**TASK 9**

**Do’s and Don'ts while working in Sprint (3-points each) ?**

**Do’s**

1. Stick to the Sprint Goal

2. Track Progress

3. Break Down Work

**Dont’s**

1. Don't Add New Work Mid-Sprint

2. Don’t Ignore Bugs or Tech Debt

3. Don’t Skip Meetings

**TASK 10**

**What are Stories and Backlogs in Scrum World ?**

**Stories -** A Story is a short, simple description of a feature or requirement told from the end-user’s perspective.

**Backlogs -** A Backlog is a prioritized list of work items (usually user stories, bugs, tasks, etc.) that need to be completed.

**TASK 11**

**Scrum Artifacts - Product Backlog, Sprint Backlog,Burn-Down Chart,**

* **Increment**

**Scrum Artifacts** - They are key pieces of information that help the team and stakeholders understand the product being developed, what work is planned, and how progress is tracked.

1. **Product Backlog** - A dynamic, ordered list of everything that is needed in the product. It includes features, enhancements, bug fixes, and technical tasks.
2. **Sprint Backlog** - A subset of the Product Backlog that the team selects to work on during a Sprint, along with a plan to deliver it.
3. **Burn-Down Chart** - A visual representation of remaining work in the Sprint (or Product Backlog) over time.
4. **Increment** - The sum of all completed Product Backlog items at the end of a Sprint — essentially the working product version.

**TASK 12**

**Ports and Protocols**

**Ports** - A port is like a door or channel through which data enters or leaves a computer over a network.

**Protocols** - A protocol is a set of rules that define how data is formatted and transmitted over a network.

**TASK 13**

**Different Network Types**

1. PAN (Personal Area Network)

2. LAN (Local Area Network)

3. MAN (Metropolitan Area Network)

4. WAN (Wide Area Network)

**TASK 14**

**Types of servers?**

**1. Web Server**

* **Function**: Hosts websites and delivers web pages to users
* **Protocols**: HTTP, HTTPS
* **Examples**: Apache, Nginx, Microsoft IIS
* **Use Case**: When you visit www.google.com, a web server sends you the content

### **2. Database Server**

* **Function**: Stores and manages databases
* **Software**: MySQL, PostgreSQL, Oracle, SQL Server
* **Use Case**: When an app retrieves user data or orders from a database

### **3. File Server**

* **Function**: Stores and shares files across a network
* **Protocols**: FTP, SMB, NFS
* **Use Case**: File sharing in an office environment

### **4. Mail Server**

* **Function**: Sends, receives, and stores email
* **Protocols**: SMTP (send), IMAP/POP3 (receive)
* **Use Case**: Gmail or Outlook backend systems

### **5. Application Server**

* **Function**: Hosts and runs specific applications or business logic
* **Examples**: Tomcat (Java), WebLogic, GlassFish
* **Use Case**: Backend for web apps (e.g., booking system)

### **6. DNS Server (Domain Name System)**

* **Function**: Translates domain names (e.g., google.com) into IP addresses
* **Use Case**: Letting users access websites by name instead of numbers

### **7. Proxy Server**

* **Function**: Acts as a gateway between users and the internet
* **Benefits**: Privacy, caching, content filtering
* **Use Case**: Used in schools or businesses to filter web access

### **8. Virtual Server**

* **Function**: Simulates a physical server using virtualization
* **Technology**: VMware, VirtualBox, Hyper-V
* **Use Case**: Running multiple server environments on one machine

### **9. Game Server**

* **Function**: Hosts multiplayer online games
* **Use Case**: Fortnite, Minecraft, or other online multiplayer platforms

### **10. Print Server**

* **Function**: Manages printers and print requests over a network
* **Use Case**: Office printer sharing across multiple employees

**TASK 15**

**DNS (Domain Name System)**

It's often called the “Phonebook of the Internet**”**, because it translates human-readable domain names (like google.com) into IP addresses (like 142.250.190.14), which computers use to communicate.

**TASK 16**

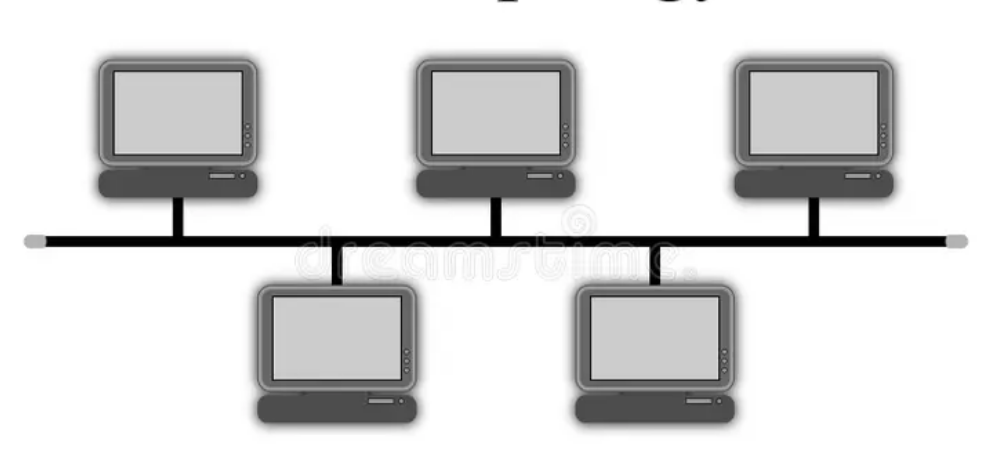
**Network Topology and its Types**

**Network Topology** - Network Topology is the layout or structure of how computers, cables, and other devices are connected in a network.

**Types of Network Topology**

### **1. Bus Topology**

* All devices are connected to a single cable (called the bus).
* Data travels in both directions along the cable.

****

**Advantages:**

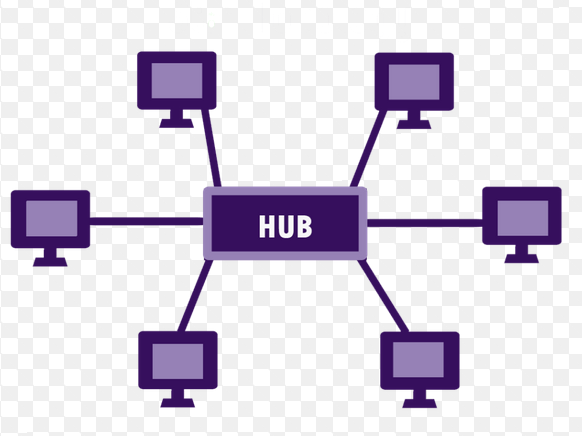
* Easy to set up and use
* Requires less cable than other topologies
* Cost-effective for small networks

**Disadvantages:**

* One cable failure can shut down the entire network
* Slow performance as more devices are added
* Hard to troubleshoot problems

### **2. Star Topology**

* All devices connect to a central device (like a switch or hub).
* Most common in home and office networks.



**Advantages:**

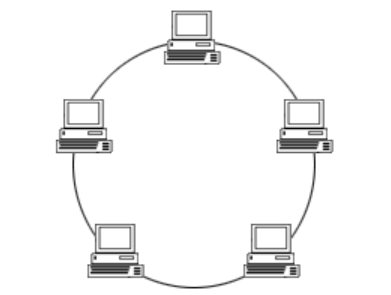
* Easy to install and manage
* If one device fails, the rest still work
* Easy to add or remove devices

**Disadvantages:**

* If the central hub/switch fails, the whole network goes down
* Uses more cable than bus topology
* Slightly more expensive due to the central device

### **3. Ring Topology**

* Devices are connected in a circle (ring).
* Data moves in one direction around the ring.



**Advantages:**

* Data flows in one direction, reducing collisions
* Performance is better than bus in heavier traffic

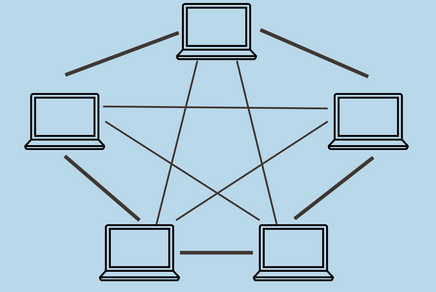
**Disadvantages:**

* One device or cable failure can affect the entire network
* Difficult to add or remove devices
* Troubleshooting is not easy

### 

### **4. Mesh Topology**

* Every device is connected to every other device.
* Used in high-security or mission-critical networks.



**Advantages:**

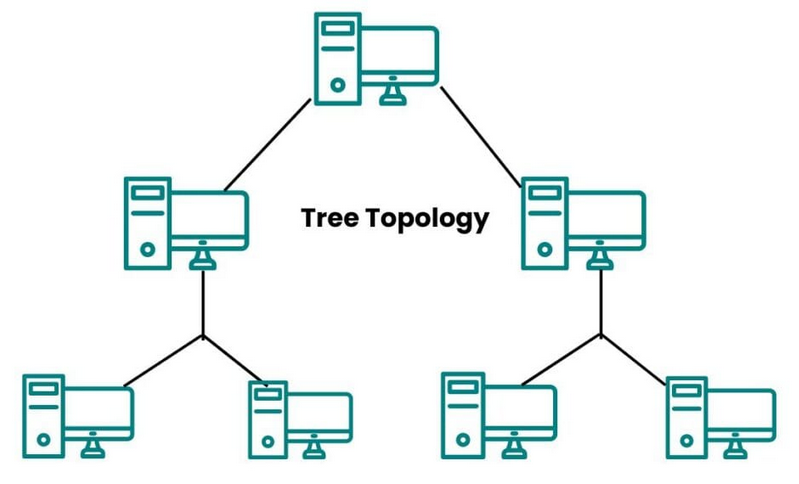
* Very reliable (multiple paths for data)
* High fault tolerance – if one link fails, data finds another way
* Secure and private

**Disadvantages:**

* Expensive to build (lots of cables)
* Complex to install and maintain
* Requires more hardware

### **5. Tree Topology**

* A mix of star and bus topologies.
* Devices are connected in a hierarchical manner.



**Advantages:**

* Easily expandable
* Well-structured and supports future growth
* Combines features of star and bus

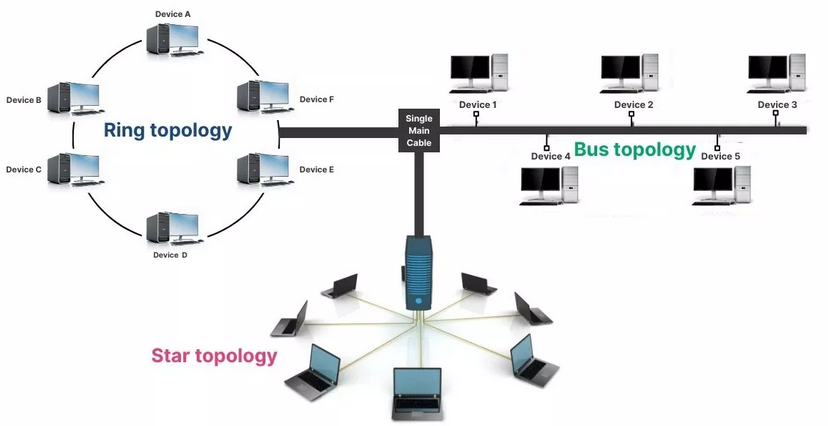
**Disadvantages:**

* If the main (root) cable fails, the whole network might stop
* More cable and planning needed
* Troubleshooting can be hard in large setups

### 

### **6. Hybrid Topology**

* A combination of two or more topologies.
* Used when different parts of an organization have different needs.



**Advantages:**

* Highly flexible and scalable
* Can be customized as per network needs
* Offers better performance and reliability

**Disadvantages:**

* Complex to design
* Costly to set up and maintain
* Needs skilled management

**TASK 17**

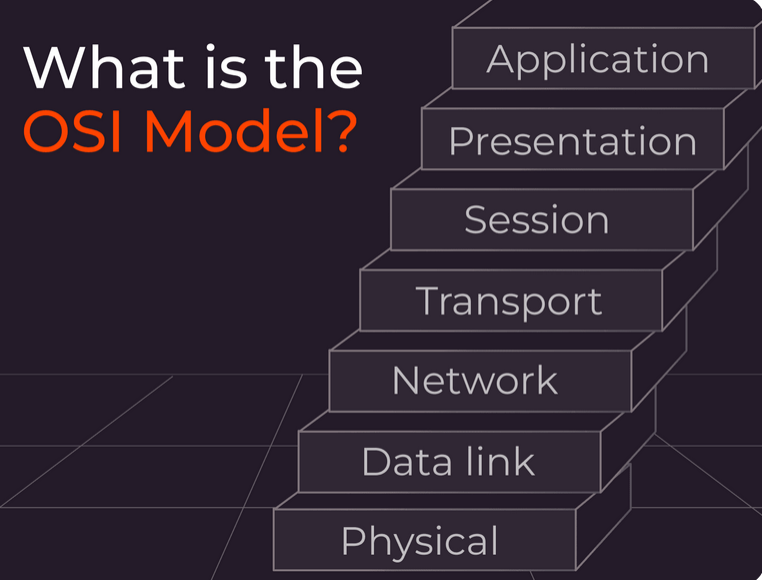
**What is OSI Model?**

**Describe the 7 layers with description.**

**OSI** stands for **Open Systems Interconnection** model.

It is a conceptual framework used to understand how data moves through a network, from one device to another.

**7 Layers of the OSI Model**

****

### 

### **1. Physical Layer (Layer 1)**

* Deals with: Hardware, cables, signals
* Job: Transfers raw bits (0s and 1s) over the network
* Examples: Ethernet cables, Wi-Fi signals, USB

### **2. Data Link Layer (Layer 2)**

* Deals with: Device-to-device communication
* Job: Breaks data into frames, handles MAC addresses, error detection
* Examples: Switches, MAC address, Ethernet

### **3. Network Layer (Layer 3)**

* Deals with: Routing the data
* Job: Chooses the best path for data, handles IP addresses
* Examples: Routers, IP (Internet Protocol)

### **4. Transport Layer (Layer 4)**

* Deals with: Data delivery and reliability
* Job: Breaks data into smaller parts (segments), checks errors
* Protocols: TCP (reliable), UDP (faster, less reliable)

### **5. Session Layer (Layer 5)**

* Deals with: Communication sessions
* Job: Starts, manages, and ends sessions between devices
* Example: You logging into a remote server and staying connected

### **6. Presentation Layer (Layer 6)**

* Deals with: Data translation
* Job: Converts data into a format readable by the application, handles encryption/decryption, compression
* Examples: JPEG, MP3, SSL, encryption

### **7. Application Layer (Layer 7)**

* Deals with: User interaction
* Job: Provides services to the user (your browser, email, apps)
* Examples: Chrome, Gmail, WhatsApp, FTP, HTTP