Software Engineering

Assignment

Module:- 1

SE – Overview of IT Industry

1)Explain in your own words what a program is and how it functions.

→A program is a set of instructions written in a programming language that tells a computer what to do. It functions by executing these instructions step by step to perform a specific task, such as calculating numbers, displaying text, or managing data.

When a program runs, the computer's processor follows the instructions, interacts with memory and storage, and produces the desired output. Programs can be simple (like a calculator) or complex (like a web browser), and they help automate tasks, solve problems, and enhance efficiency in various fields

2) Explain in your own words what a program is and how it functions. What is Programming?

Ans:- A computer program is nothing but a set of instructions (smallest unit of execution) that are used to execute particular tasks to get particular results.

What are Functions in Programming

-→Functions in Programming is a block of code that encapsulates a specific task or related group of tasks.

3) What are the key steps involved in the programming process?

→Problem Definition – Understand and define the problem clearly.

Planning & Algorithm Design – Outline the logic and structure using flowcharts or pseudocode.

Coding – Write the actual program using a programming language.

Compilation & Execution – Convert the code into machine language and run it.

Testing & Debugging – Identify and fix errors through testing.

Documentation – Record details for future reference and maintenance.

Maintenance & Updates – Modify and improve the program as needed.

4) What are the main differences between high-level and low-level programming languages?

high-level	low-level
1) high-level s less memory efficient.	1) low-level is high memory efficient
2) It is easy to understand.	2) It is tough to understand.
3) Debugging is easy.	3) Debugging is complex comparatively.
4) It is simple to maintain.	4) It is complex to maintain comparatively.
5) It is portable.	5)It is non-portable.
6) It can run on any platform.	6) It is machine-dependent.
7)It needs compiler or interpreter for translation.	7) It needs assembler for translation.
8) It is used widely for programming.	8) It is not commonly used now-a-days in programming.
9) Java,Python,c++	9) Assembler language

4) Describe the roles of the client and server in web communication.

LAB EXERCISE: Research and create a diagram of how data is transmitted rom a client to a server over the internet.

→ Web communication follows the client-server model, where both the client and server play specific roles in exchanging information over a network.

Client Role:

The client is typically a web browser, mobile app, or software that requests resources or services from the server.

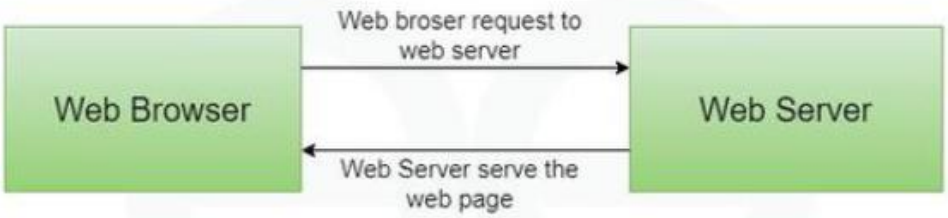
- Sends HTTP/HTTPS requests to the server.

- Displays web pages or application data received from the server.
- Can include **caching mechanisms** to store data for faster access.
- May run client-side scripts for dynamic content.

Server Role:

The server is a powerful computer or cloud service that listens for client requests and responds accordingly.

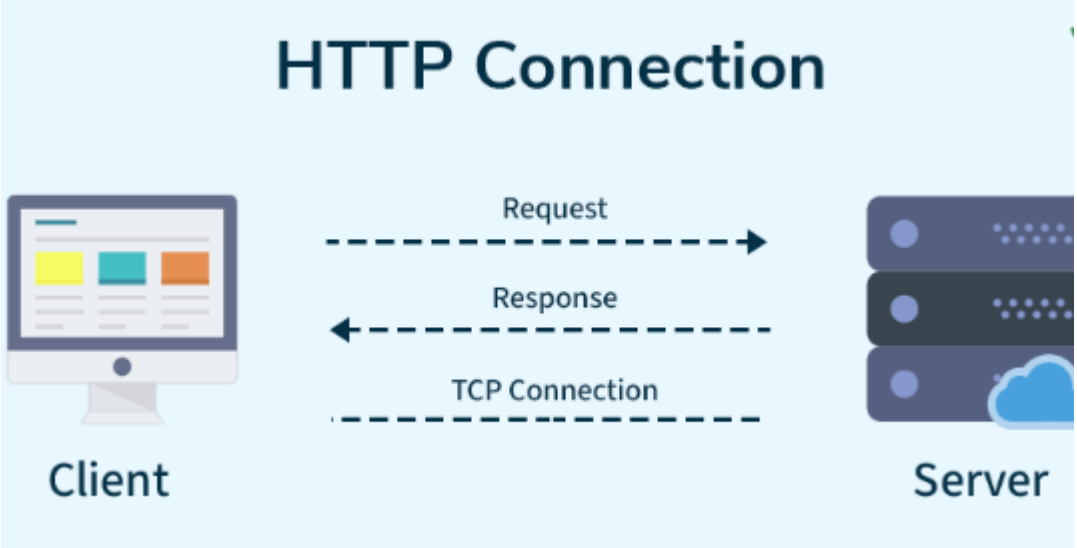
- Processes incoming HTTP/HTTPS requests.
- Fetches or manipulates data from a database.
- Runs server-side scripts (e.g., PHP, Node.js, Python, etc.).
- Sends HTML, CSS, JavaScript, or JSON responses back to the client.
- Implements security measures .



5) Explain the function of the TCP/IP model and its layers.

LAB EXERCISE: Design a simple HTTP client-server communication in any language.

→



1) **Client Role**

- The client sends a request to the server (e.g., opening a webpage).
- It waits for the server's response.
- Once the response arrives, the client processes and displays the data.

2) Server Role

- The server listens for requests from the client.
- It processes the request (e.g., fetching data from a database).
- Then, it sends a response back to the client.

Example of Communication

1. The client requests a webpage by entering www.example.com in a browser.
2. The server receives the request and retrieves the webpage.
3. The server sends the webpage data back to the client.

4.The client displays the webpage to the user.

→ The TCP/IP (Transmission Control Protocol/Internet Protocol) model is a set of rules that allows computers and devices to communicate over the internet. It ensures that data is transmitted, routed, and received correctly across networks.

Layers of the TCP/IP Model

1. Application Layer

- Handles communication between applications (e.g., web browsers, email).
- Uses protocols like HTTP, HTTPS, FTP, SMTP, DNS.

- Example: A web browser requests a webpage using HTTP.

2. Transport Layer

- Ensures reliable data delivery between devices.
- Uses two main protocols:
    - TCP (Transmission Control Protocol): Reliable, ordered, and error-checked delivery (e.g., web pages, emails).
    - UDP (User Datagram Protocol): Faster but unreliable (e.g., live streaming, online gaming).

3. Internet Layer

- Manages routing of data across networks.
- Uses IP (Internet Protocol) to assign addresses to devices.
- Ensures data packets reach the correct destination.

4. Network Access Layer (Link Layer)

- Handles physical network connections (e.g., Wi-Fi, Ethernet).
- Converts data into signals (electrical, radio, etc.) for transmission.

6) **Explain Client Server Communication?**

→ Client-Server Communication

Client-server communication is the exchange of data between a client (user device) and a server (central computer) over a network. The client **requests** services, and the server **responds** by providing the required data or services.

How It Works

1. **Client Sends a Request** – A user enters a website URL in a browser.
2. **Server Processes the Request** – The server retrieves or processes the required data.
3. **Server Sends a Response** – The server sends the webpage or data back to the client.
4. **Client Displays the Data** – The browser loads and displays the webpage.

Examples of Client-Server Communication

- A web browser accessing a website.
- A mobile app retrieving data from an online database.
- A video streaming service providing content on demand.

7) How does broadband differ from fiber-optic internet?

LAB EXERCISE: Research different types of internet connections (e.g., broadband, fiber, satellite)and list their pros and cons.

→ Broadband (DSL/Cable) Protocols: Uses PPPoE (Point-to-Point Protocol over Ethernet) for DSL and DOCSIS (Data Over Cable Service Interface Specification) for cable internet.

Fiber-Optic Protocols: Uses GPON (Gigabit Passive Optical Network) and EPON (Ethernet Passive Optical Network) for data transmission.

Key Difference: Broadband transmits data using electrical signals over copper cables, while fiber-optic internet uses light signals through fiber-optic cables, making it faster and more reliable.

LAB EXERCISE: Research different types of internet connections (e.g., broadband, fiber, satellite)and list their pros and cons.

→ **Broadband (DSL/Cable)**

Widely available, affordable

Slower than fiber, speed varies

Fiber-Optic

Very fast, reliable, low latency

Expensive, limited coverage

Satellite

Works in remote areas, no cables

High latency, weather issues

Mobile (3G/4G/5G)

Wireless, portable, fast (5G)

Data limits, signal-dependent

Fixed Wireless

No cables, good for rural areas

Affected by weather, slower than fiber

8) What are the differences between HTTP and HTTPS protocols?

LAB EXERCISE: Simulate HTTP and FTP requests using command line tools (e.g., curl).

→ **1. HTTP Requests (Web Communication)**

- Used to access websites and APIs.
- A browser usually makes these requests when you visit a webpage.
- You can send requests to get, send, or update data.

Example:

- A request is sent to a website to load a page.
- The server processes the request and sends back the webpage.

2. FTP Requests (File Transfer Protocol)

- Used to upload and download files from a remote server.
- Commonly used for website hosting and backups.

Example:

- You request to download a file from a server.
- The server sends the file to your computer.

HTTP	HTTPS
HTTP stands for HyperText Transfer Protocol.	HTTPS for HyperText Transfer Protocol Secure.
HTTP, URL begins with “http://”.	HTTPs, URL starts with “https://”.
HTTP uses port number 80 for communication.	HTTPs uses 443 port number for communication.
HTTP is considered to be unsecure.	HTTPs is considered as secure.
HTTP works at Application Layer.	HTTPS works at Transport Layer.
HTTP faster than HTTPS	HTTPS slower than HTTP
Search engines do not favour the insecure website.	Improved reputation of the website in search engine.

9) What is the role of encryption in securing applications?

LAB EXERCISE: Identify and explain three common application security vulnerabilities. Suggest possible solutions.

→Encryption plays a critical role in securing applications by transforming sensitive data into an unreadable format, accessible only to authorized users with the decryption key.

Software Applications and Its Types: Software applications are programs designed to perform specific tasks for users.

1. System Software

- Manages hardware and system resources.
- Examples: Operating systems like Windows, macOS, and Linux.

2. Application Software

- Designed for end-users to perform specific tasks.
- Examples: Microsoft Word (word processing), Excel and web browsers.

3. **Development Software**

Provides tools for software developers to create applications.

Examples: IDEs like Visual Studio, PyCharm, and compilers.

4. **Middleware**

- Connects different software applications or systems to enable communication.
- Examples: Database management systems (DBMS), APIs, and message brokers.

10) What is the difference between system software and application software?

LAB EXERCISE:Identify and classify 5 applications you use daily as either system software orapplication software.

→

Application	Type	Reason
Android OS	System Software	Manages hardware resources and supports other apps.
Google Maps	Application Software	Provides navigation and location services.
WhatsApp	Application Software	Used for messaging, calls, and media sharing.
Task Manager	System Software	Monitors and controls system performance and processes.
YouTube	Application Software	Streams videos and content for entertainment and learning.

System Software	Application Software
System Software maintains the system resources and gives the path for application software to run.	Application software is built for specific tasks.
Low-level languages are used to write the system software.	While high-level languages are used to write the application software.
Example: System software is an operating system, etc.	Example: Application software is Photoshop
System Software programming is more complex than application software.	Application software programming is simpler in comparison to system software.
System software runs independently.	Application software is dependent on system software because they need a set platform for its functioning.

11) What is the significance of modularity in software architecture?

→ Maintainability – Easy updates and fixes.

Reusability – Use modules in different projects.

Scalability – Add new features easily.

Debugging – Isolate and fix errors faster.

Collaboration – Teams work independently.

Flexibility – Modify modules with minimal impact.

Performance – Optimized modules improve efficiency.

Faster Development – Reuse saves coding time.

LAB EXERCISE: Design a basic three-tier software architecture diagram for a web application.

The Three-Tier Client-Server Architecture divides systems into presentation, application, and data layers, increasing scalability, maintainability, and efficiency.

Architecture design model:

- **Presentation Tier:** The user interface layer, where interactions occur. It handles data display and user input.

- **Application Tier**:The business logic layer, which processes user requests, performs computations, and makes decisions. It acts as a mediator between the presentation and data tiers.

Data Tier:The storage layer, is responsible for managing and storing data. It handles database operations and data retrieval

three-tier software architecture

12) Why are layers important in software architecture?

→ Separation of Concerns: Each layer handles a specific responsibility, improving code organization.
Scalability: Easier to modify or expand individual layers without affecting the entire system.
Maintainability: Clear structure simplifies debugging, testing, and future updates.
Reusability: Common logic (e.g., authentication, logging) can be reused across multiple applications.
Flexibility: New technologies or features can be integrated with minimal disruption

13) Explain the importance of a development environment in software production.

LAB EXERCISE: Explore different types of software environments (development, testing, production).Set up a basic environment in a virtual machine.

→ **Error Detection:** Early identification of syntax, logic, and runtime errors prevents issues from progressing.

Version Control: Enables tracking of code changes and collaborative development.
Dependency Management: Ensures all required libraries, frameworks, and tools are correctly configured.

Automated Testing: Developers can run unit tests frequently to validate functionality.
Customization: Allows developers to configure environments to match production settings

closely.
Debugging Support: Tools like breakpoints, watch variables, and stack traces help resolve issues faster.
Safe Experimentation: New features can be explored without risking stability.
Efficient Collaboration: Teams can work on different features independently and merge their changes smoothly.

Types of Software Environments

1. Development Environment – Used for coding and initial testing.
2. Testing Environment – Runs tests to find bugs before deployment.
3. Production Environment – Live system used by end-users.

Setting Up a Basic Environment in a Virtual Machine

1. Install Virtual Machine Software (e.g., VirtualBox, VMware).
2. Create a New VM and select the OS.
3. Allocate CPU, RAM, and Storage.
4. Install the OS (Linux/Windows).
5. Set up Development Tools (e.g., IDEs, databases, servers).
6. Configure Networking & Security settings.
7. Test the environment to ensure proper functionality.

14) What is the difference between source code and machine code?

LAB EXERCISE: Write and upload your firstsource code file to Github.

Source Code	Machine Code
Human-readable instructions written in programming languages (e.g., Python, Java).	Binary instructions (0s and 1s) that a computer's CPU can directly execute.
Easily understandable by developers.	Difficult for humans to read or interpret.
Written using text-based syntax (e.g., if, for, print).	Consists of binary code or hexadecimal values.
Requires compilation or interpretation to become executable.	Directly executed by the computer's hardware.
print("Hello World") in Python.	10110000 01100001 (Binary equivalent of machine instructions).

LAB EXERCISE: Write and upload your firstsource code file to Github.

Create a Code File → Write and save a simple program (e.g., firstcode.py).

Create a GitHub Repository → Go to GitHub, click **New repository**, and create one.

Upload File

- **Via Website**: Click **Add file** → **Upload files** → **Commit changes**.
- **Via Git (Command Line)**:

sh

git init

git add firstcode.py

git commit -m "Initial commit"

git remote add origin <repo-URL>

git branch -M main

git push -u origin main

Verify Upload → Check your GitHub repository.

15) **Why is version control important in software development?**

LAB EXERCISE: Create a student account on Github and collaborate on a small project with aclassmate.

→ Tracks Changes → Keeps a history of modifications, allowing developers to revert to previous versions if needed.

Collaboration → Enables multiple developers to work on the same project without overwriting each other's changes.

Backup & Recovery → Protects code from accidental loss or corruption.

Branching & Merging → Supports parallel development by allowing new features or fixes to be developed separately and merged later.

Code Integrity → Helps in identifying and resolving conflicts between different versions of code.

16) What are the benefits of using Github for students?

→Benefits of GitHub for Students

1. GitHub Student Pack – Free access to premium tools.
2. Version Control & Collaboration – Work efficiently with teams.
3. Portfolio Building – Showcase projects for job opportunities.
4. Open Source Contributions – Learn and network with professionals.
5. Code Backup & Management – Secure cloud storage & version history.
6. Learning & Community Support – Gain insights from experienced developers.
7. Tool Integration – Connect with IDEs, CI/CD, and cloud services.
8. Private Repositories – Work on personal projects securely.

17) What the role of application software in businesses?is

LAB EXERCISE: Write a report on the various types of application software and how they improveproductivity.

→ Automation of Tasks: Tools like Excel automate data analysis, reducing manual effort.
Data Management: CRM systems (e.g., Salesforce) organize customer data and improve decision-making.
Enhanced Communication: Platforms like Microsoft Teams and Slack enable seamless collaboration.
Financial Management: Accounting software like QuickBooks helps manage budgets, invoices, and payroll.
Customer Support: Helpdesk software like Zendesk improves client interaction.

18) Create a flowchart representing the Software Development Life Cycle (SDLC).

→ A step by step approach to develop any product/software with high quality, lowest cost with Shortest possible time.

1. Requirement Analysis

- Purpose: Understand user needs and system requirements.
- Key Activities: Gather requirements, document them, and get stakeholder approval.

2. Planning

- Purpose: Define project scope, timelines, and resources.
- Key Activities: Create a project plan, set milestones, and allocate tasks.

3. Design

- Purpose: Outline the software's architecture and technical specifications.
- Key Activities: Design UI/UX, database models, and system structure.

4. Development (Coding)

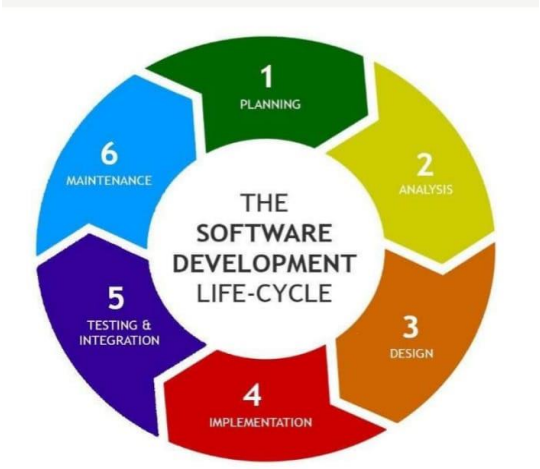
- Purpose: Write and implement the actual code based on the design.
- Key Activities: Develop features, integrate APIs, and ensure proper functionality.

5. Testing

- Purpose: Identify and fix bugs to ensure software reliability.
- Key Activities: Perform unit testing, integration testing, and user acceptance testing (UAT).

7. Maintenance & Support

- Purpose: Address issues, provide updates, and improve performance.
- Key Activities: Bug fixes, security updates, and feature enhancements.



19) What are the main stages of the software development process?

→ The main stages of the software development process, also known as the Software Development Life Cycle (SDLC), generally include planning, requirements analysis, design, coding, testing, deployment, and maint**enance.**

20) Write a requirementspecification for a simple library managementsystem.

→ Requirements analysis is a critical part of the requirements definition and management process in software development. The purpose of requirements analysis is to be sure all product requirements accurately represent stakeholder needs and requirements.

21) Why is the requirement analysis phase critical in software development? Software Analysis.

→The Requirement Analysis phase is critical in software development because it lays the foundation for the entire project**.**

1. Clear Understanding of Project Scope

- Defining what the software must achieve ensures developers know exactly what to build.
- Avoids scope creep (uncontrolled changes or continuous growth in project scope).

2. Identifying Stakeholder Needs

- Capturing the expectations of clients, end-users, and project teams ensures the final product aligns with their goals.
- In Agile, this phase involves continuous collaboration to adapt to changing requirements.

3. Improved Planning and Estimation

- Well-defined requirements help in estimating timelines, costs, and resource allocation accurately.
- Reduces the risk of budget overruns.

4. Reducing Rework

- Identifying potential challenges early minimizes future modifications.
- Clear requirements lead to smoother design, coding, and testing stages.

5. Ensuring Functional and Non-Functional Needs Are Met

- Functional requirements describe what the system should do (e.g., login functionality).

- Non-functional requirements focus on how the system performs (e.g., speed, security).

6. Basis for Test Case Development

- Testers use requirement documents to create comprehensive test cases that cover all expected scenarios.

22) What is the role of software analysis in the development process?

→Requirement Gathering: Collects and documents user needs through meetings, interviews, etc.

Requirement Validation: Ensures requirements are complete, consistent, and feasible.

Defining System Scope: Establishes project boundaries to prevent scope creep.

Creating Documentation: Produces key documents like SRS, Use Cases, and User Stories.

Risk Identification: Detects potential technical challenges early.

Supporting Design & Development: Translates user needs into technical requirements.

Facilitating Communication: Bridges the gap between technical teams and stakeholders.

23) What are the key elements of system design?

→Architecture Design: Defines system structure (e.g., client-server, microservices).

Module Design: Divides the system into manageable components.

Data Design: Focuses on database structure, schemas, and data flow.

Interface Design: Covers UI/UX and API integration.

Security Design: Ensures data protection with encryption and authentication.

Performance Design: Optimizes system speed and resource usage.

Error Handling Design: Manages exceptions for smooth recovery.

Scalability & Reliability Design: Ensures system growth and stability under load.

24) Why is software testing important?

→Ensures Quality: Verifies that the software meets functional and non-functional requirements.

Detects Bugs Early: Identifies issues during development, reducing costly fixes later.

Enhances Security: Uncovers vulnerabilities to protect user data and prevent breaches.

Improves Performance: Ensures the software runs efficiently under various conditions.

Boosts User Satisfaction: Provides a smooth and error-free user experience.

Ensures Compatibility: Confirms the software works across different devices, browsers, and platforms.

Validates Functionality: Ensures all features behave as intended.

Reduces Maintenance Costs: Early issue detection minimizes future maintenance expenses.

25) What types of software maintenance are there?

→Corrective Maintenance: Fixes bugs and errors.

Adaptive Maintenance: Updates software for changing environments.

Perfective Maintenance: Adds new features or improves performance.

Preventive Maintenance: Enhances code to prevent future issues.

26) What are the key differences between web and desktop applications?

→

Aspect	**Web Application**	**Desktop Application**
Access	Accessed via web browsers (e.g., Chrome).	Installed and run directly on a device.
Internet Requirement	Requires an internet connection	Requires an internet connection
Installation	No installation needed; accessed via URL.	Requires installation on each device.
Updates	Updates are automatic on the server-side.	Users must manually update the software.

Performance	Performance may depend on internet speed.	Typically faster since it runs locally.

27) What are the advantages of using web applications over desktop applications?

→ Accessibility: Use from any device with a browser.

No Installation: Access directly via URL.

Automatic Updates: Always up-to-date.

Cross-Platform Support: Works on multiple OS.

Lower Maintenance Cost: Centralized updates reduce effort.

Easy Collaboration: Supports real-time teamwork.

Secure Data Storage: Data stored safely in the cloud.

Scalability: Easily handles growing user demands.

28) What role does UI/UX design play in application development?

→ User Engagement: Creates visually appealing and interactive interfaces.

Improved Usability: Ensures intuitive navigation and user-friendly layouts.

Enhanced Accessibility: Designs inclusive experiences for all users.

Customer Satisfaction: Boosts user retention with smooth experiences. Reduced Support Needs: Clear design minimizes user confusion.

Stronger Brand Identity: Consistent design reinforces brand image.

Increased Conversion Rates: Optimized designs guide users toward desired actions.

29) What are the differences between native and hybrid mobile apps?

→

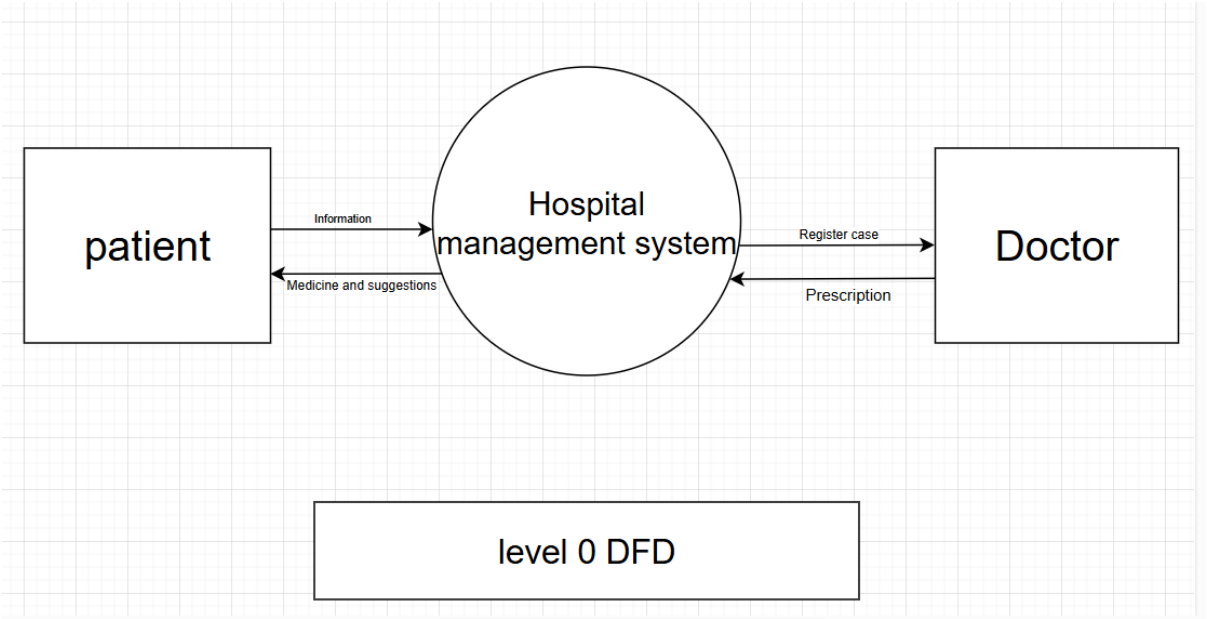
Feature	**Native Apps**	**Hybrid Apps**
Development Speed	Slow	Fast
Maintenance Cost	High	Low
Graphical Performance	Very High	Moderate

Feature	Native Apps	Hybrid Apps
Language Used	Kotlin, Java, Swift	HTML, CSS, JavaScript
Code Portability	Tough	Easy

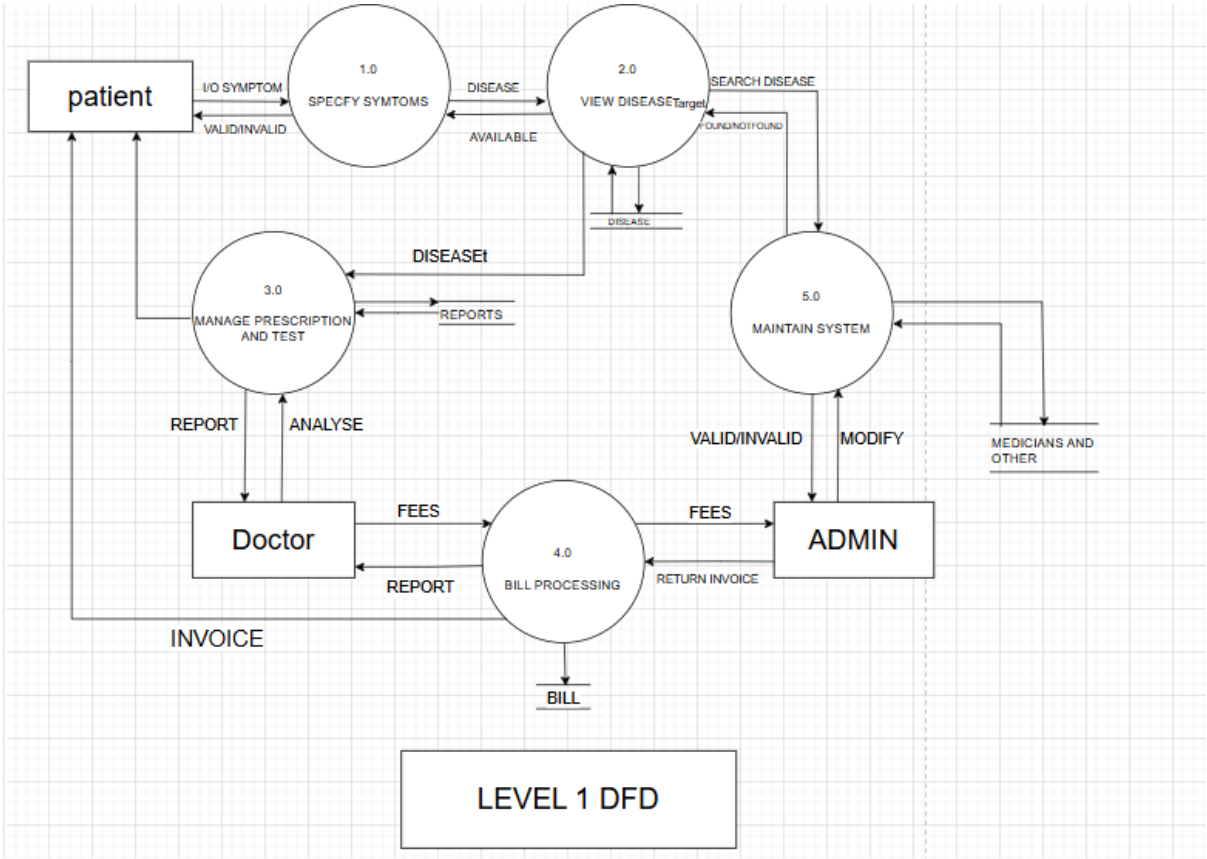
30) What is the significance of DFDs in system analysis?

Create a DFD for a hospital management system

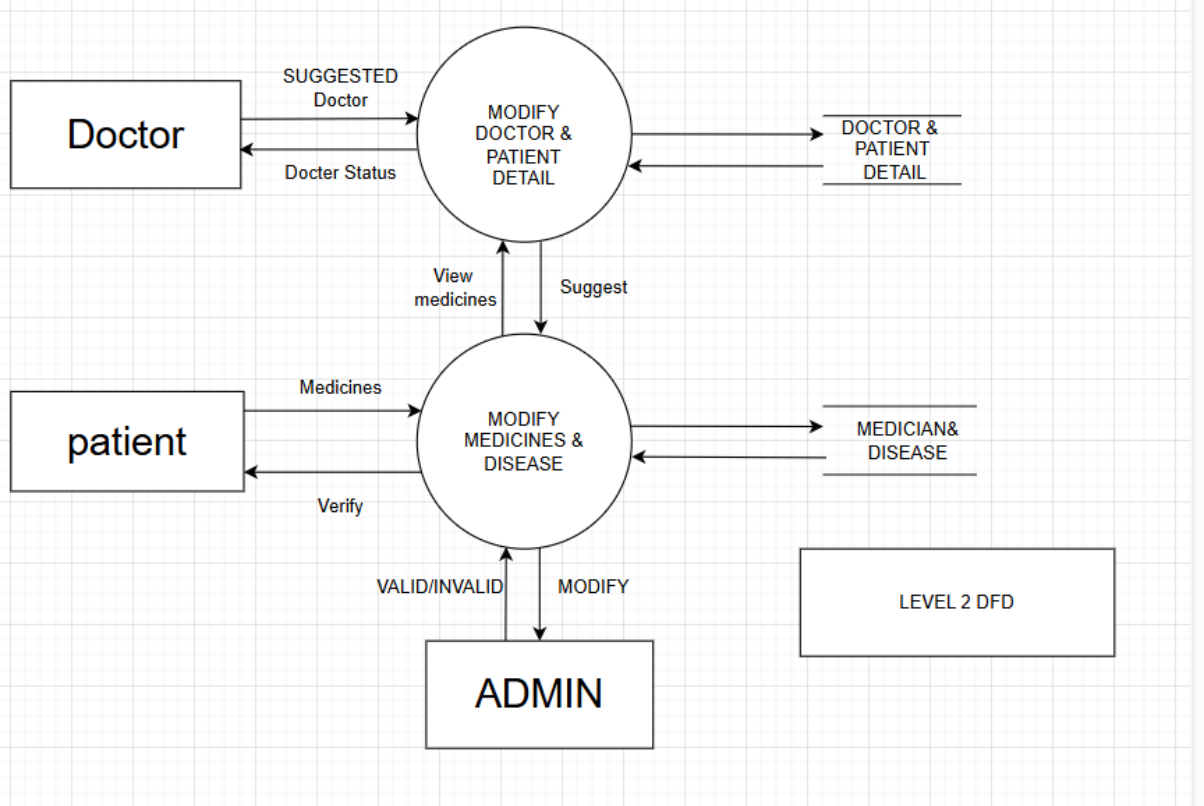
→Data Flow Diagrams (DFDs) are crucial in system analysis as they visually represent how data moves within a system. They help identify processes, data sources, and storage, making complex systems easier to understand. DFDs aid in detecting inefficiencies and improving system design.



level 0 DFD



LEVEL 1 DFD



31) What are the pros and cons of desktop applications compared to webapplications?

→

Feature	Desktop Applications	Web Applications
Performance	Faster, uses local resources.	Can be slower due to internet dependency.
Access	Works offline.	Requires an internet connection.
Security	More secure, as data is stored locally.	Data is stored on servers, vulnerable to cyberattacks.
Updates	Manual installation required.	Automatic updates, no user action needed.
Compatibility	OS-dependent (Windows, macOS, etc.).	Works across all operating systems.

32)How do flowcharts help in programming and system design?

Draw a flowchart representing the logic of a basic online registration system

→

1)Visual Clarity – Simplifies complex logic with diagrams.
2)Problem-Solving – Helps in debugging and identifying errors.
3)Efficient Planning – Aids in designing algorithms before coding.
4)Improved Communication – Makes it easier for teams to understand logic.
5)Standardized Representation – Uses universal symbols for clarity.
6)Better Documentation – Serves as a reference for future development.