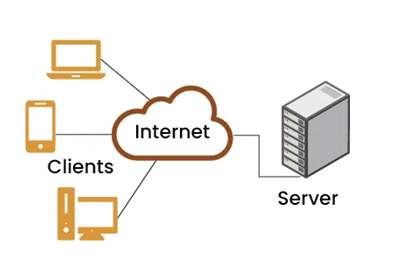
* **Theory Assignment-1**

# Module -1

**Q-1 Research and create a diagram of how data is transmitte from a client to a server over the internet.**

**Ans :-**

In a client-server model, data transmission over the internet involves a client initiating a request, which is then processed server, and a response being sent back to the client.



[Daigram-client architecture]

In detail:-

Step 1: User Enters a URL

The user opens a web browser and types in a website address like www.example.com.

Step 2: DNS Resolution

The browser contacts a DNS (Domain Name System) server to find the IP address of the website. This helps the computer locate the correct server.

Step 3: Establishing a Connection

Once the IP address is known, the client starts a connection with the server using TCP (Transmission Control Protocol), which ensures the connection is stable and reliable.

Step 4: Securing the Connection (if HTTPS)

If the website uses HTTPS, a secure connection is created using TLS (Transport Layer Security). This makes sure the data stays private and protected.

Step 5: Sending a Request

The client sends an HTTP request to the server. This request asks for the content of the webpage (like text, images, etc.).

Step 6: Data Travels Through the Internet

The request travels through various routers and networks until it reaches the correct server.

Step 7: Server Processes the Request

The server receives the request, understands what is needed, and prepares a response (usually a webpage).

Step 8: Server Sends Back Data

The server sends the requested data back to the client in small packets.

Step 9: Browser Displays the Website

The client’s browser puts all the packets together and shows the final webpage to the user.

**Q-2 Explain the function of the TCP/IP model and its layers.**

**Ans:-**

Function of the TCP/IP Model-

The TCP/IP model (Transmission Control Protocol/Internet Protocol) is a set of rules that allows computers to communicate over the internet. It helps organize how data is sent and received between devices.

The model breaks down communication into layers, where each layer has a specific job. This makes it easier to manage and troubleshoot network problems.

* Layers of the TCP/IP Model

The TCP/IP model has 4 main layers:

1. Application Layer

Function: This is where communication starts. It provides services like email, web browsing, file transfer, etc.

Examples: HTTP, FTP, SMTP, DNS

User Interaction: This is the layer closest to the user.

2. Transport Layer

Function: Manages the delivery of data between devices. It breaks data into smaller pieces and ensures it arrives correctly.

Main Protocols:

TCP (Transmission Control Protocol): Reliable and error-checked.

UDP (User Datagram Protocol): Faster but less reliable.

Features: Error checking, data flow control.

3. Internet Layer

Function: Chooses the best path for data to travel and moves it between networks.

Main Protocol: IP (Internet Protocol)

Job: Adds IP addresses and routes the data packets.

4. Network Access Layer (Link Layer)

Function: Handles how data is physically sent over cables or wireless.

Includes: Device drivers, network interface cards (NICs), Ethernet, WiFi

Job: Moves data within the local network.

**Q-3 Explain Client Server Communication .**

**Ans:-**

Client-server communication is a fundamental network model where a client (e.g., a web browser, an app) requests resources or services from a server (e.g., a web server, a database server). The server processes the request and sends back the requested data or performs the requested action. This interaction is a request-response cycle, with the client initiating the communication and the server responding.

Key Components and Process:

Client:

A computer or software application that initiates the communication by sending requests to the server.

Server:

A computer or software application that provides resources, services, or data to the client.

Network:

The communication channel (e.g., internet, local network) that connects the client and server.

Request:

The client sends a request to the server, specifying what it needs (e.g., a webpage, a file, a database record).

Response:

The server processes the request and sends back a response, which may include the requested data or an acknowledgment of the action performed.

Example:

When you type a website address into your browser (the client), your browser sends a request to the web server hosting that website. The web server then sends back the website's files (HTML, CSS, JavaScript, images, etc.) which your browser then uses to display the webpage

Communication Protocols:

Clients and servers use specific protocols to communicate, such as:

HTTP/HTTPS: Used for web browsing and communication with web servers.

FTP: Used for file transfer.

SMTP, POP3, IMAP: Used for email communication.

TCP/IP: A set of rules that govern how data is transmitted over networks, providing reliable and ordered data delivery.

**Q-4 Research different types of internet connections (e.g., broadband, fiber,satellite)and list their pros and cons.**

**Ans:-**

Introduction

The internet has become a basic need in today’s digital world. Whether it’s for communication, education, business, or entertainment – a good internet connection is essential. But not all internet connections are the same. In this assignment, I have researched and explained different types of internet connections, including their advantages and disadvantages. This will help us understand which type is best suited for different needs.

1. Broadband (DSL and Cable)

➤ What is it?

Broadband is a high-speed internet connection. It can be delivered through DSL (Digital Subscriber Line) using telephone lines or through cable using TV cables.

✅ Pros:

* Faster than dial-up
* Always connected (no need to dial)
* Available in most cities and towns
* Affordable for most households

❌ Cons:

* Speed can reduce during peak hours
* Cable broadband is shared, so performance may vary
* DSL may be slower if you are far from the service provider

2. Fiber Optic Internet

➤ What is it?

Fiber internet uses thin glass or plastic fibers to send data at very high speeds using light signals.

✅ Pros:

* Extremely fast (up to 1 Gbps or more)
* Reliable and stable connection
* Great for streaming, gaming, and video calls
* Upload and download speeds are almost equal

❌ Cons:

* Expensive installation
* Not available in all areas (mainly in cities)
* Takes time to expand to rural areas

3. Satellite Internet

➤ What is it?

Satellite internet sends signals from a satellite in space to a dish installed at your home. It’s often used in remote or rural areas.

✅ Pros:

* Available almost anywhere on Earth
* Good for areas where other options don’t exist
* Wireless connection (no cables needed)

❌ Cons:

* Slower compared to fiber or broadband
* High latency (delay), bad for gaming or live calls
* Can be affected by weather (rain, storms)
* Expensive data plans

4. Mobile Internet (4G/5G)

➤ What is it?

Mobile internet uses your mobile phone network (4G or 5G) to connect to the internet. You can use it on your phone or hotspot it to other devices.

✅ Pros:

* Portable and wireless
* Easy to set up
* Fast speeds with 4G and 5G
* Great for travel or temporary use

❌ Cons:

* Data limits can be expensive
* Speeds vary based on location
* Battery usage is high if used as a hotspot

5. Dial-Up Internet (Old Method)

➤ What is it?

Dial-up uses a telephone line to connect to the internet. It's very slow and not common today.

✅ Pros:

* Very cheap
* Can work in areas without modern infrastructure

❌ Cons:

* Extremely slow (not suitable for modern usage)
* Can’t use phone and internet at the same time
* Outdated technology

**Q-5 Design a basic three-tiersoftware architecture diagram for a web application**

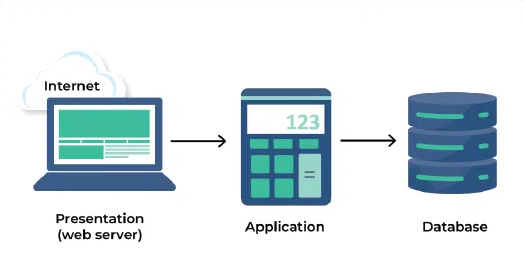
**Ans:-**

Introduction

In software engineering, a three-tier architecture is a popular design used in modern web applications. It divides the application into three separate layers, each with its own responsibilities. This makes the app easier to manage, update, and scale.

Three Tiers:

1. Presentation Layer (Client/User Interface)
2. Application Layer (Business Logic)
3. Data Layer (Database)



In detail:

**1. Presentation Tier (Client Tier/User Interface):**

**Function:**

This is the user interface (UI) layer that users directly interact with. It handles displaying information to users and collecting their input.

**Examples:**

Web browsers, mobile apps, or desktop applications.

**Technology:**

HTML, CSS, JavaScript for web applications; native UI frameworks for mobile or desktop apps.

**Interaction:**

Sends user requests to the Application Tier and displays the results received from it.

**2. Application Tier (Business Logic Tier):**

**Function:** This layer contains the core logic and functionality of the application. It processes user requests, performs calculations, and interacts with the Data Tier.

**Examples:** Servers running application code (e.g., Java, Python, Node.js).

**Technology:** Programming languages like Java, Python, Node.js, etc.

**Interaction:** Receives requests from the Presentation Tier, retrieves data from or stores data in the Data Tier, and sends results back to the Presentation Tier.

**3. Data Tier (Database Tier):**

**Function:** This layer manages and stores the application's data. It provides a persistent storage solution for the application.

**Examples:** Databases (SQL or NoSQL), data warehouses.

**Technology:** Databases like MySQL, PostgreSQL, MongoDB, etc.

**Interaction:** Stores, retrieves, and updates data based on requests from the Application Tier.

Diagram Representation:

A simple visual representation would show:

**Client (Presentation Tier):** Connected to the internet, sends requests.

**Web Server (Application Tier):** Receives requests from the client.

**Application Server (Application Tier):** Processes requests and interacts with the database.

**Database (Data Tier):** Stores and retrieves data.

**Q-6 Create a case study on the functionality of the presentation, business logic, and data access layers of a given software system.**

**Ans:-**

1. System overview

The system handles customer transactions, such as viewing account information, initiating transfers, and managing personal details. Initially, it was a monolithic application, difficult to modify and maintain. By adopting a layered architecture, the company aims to enhance the system's adaptability, scalability, and reusability.

2. Layered architecture:

presentation, business logic, and data access The refactored system utilizes a three-tiered architecture:

* Presentation layer:

This layer is responsible for the user interface and interactions. It displays information to the user and captures their input.

* Business logic layer: This is the core of the application, processing business rules and orchestrating the system's workflows. It acts as an intermediary between the presentation and data access layers.
* Data access layer: This layer manages data storage and retrieval, providing a standardized interface for the business logic layer to interact with the database.

3. Functionality and interaction:

* Presentation layer

1.Functionality: The presentation layer displays customer account summaries, transaction history, forms for initiating transfers, and personal information. It also handles user input, such as login credentials, transaction details, and updates to personal information.

2.Interaction: It sends user requests and data to the business logic layer for processing. It receives processed data and display updates from the business logic layer to provide the user with the appropriate feedback.

* Business logic layer

1.Functionality: The business logic layer orchestrates the transaction process. It validates user input (e.g., ensuring sufficient funds for a transfer), applies business rules (e.g., calculating interest on a savings account), and determines the sequence of operations required to fulfill a transaction. This layer might use domain objects and services to encapsulate the system's core functionality.

2.Interaction: It receives data and requests from the presentation layer and passes them to the data access layer for storage or retrieval. It receives data from the data access layer, processes it according to business rules, and sends the results back to the presentation layer for display.

1. Benefits realized:

1.Improved Modularity: Each layer is responsible for a specific concern, making the system easier to understand, manage, and maintain. Changes in one layer have a minimal impact on others.

2.Enhanced Scalability: Individual layers can be scaled independently to handle increased load. For example, the presentation layer might be scaled to accommodate more concurrent users, or the data access layer might be scaled to handle more database queries.

3.Increased Reusability: Components within a layer can be reused across different parts of the application or even in different projects, reducing development effort and promoting consistency.

4.Simplified Testing: Each layer can be tested in isolation, simplifying the testing process and making it easier to identify and fix issues.

Better Adaptability: Changes in presentation technology or database systems can be accommodated more easily without requiring extensive modifications to the entire application.

**Q-7 Explore different types of software environments (development, testing,production).Set up a basic environment in a virtual machine.**

**Ans:-**

Introduction

In software development, environments play a crucial role in the lifecycle of an application. The three primary types of environments are development, testing, and production. Each serves a distinct purpose and has its own characteristics. This assignment will explore these environments and guide you through setting up a basic environment in a virtual machine (VM).

* Types of Software Environments:

1. Development Environment

The development environment is where developers write and test their code. It is typically set up on local machines or dedicated servers. Key characteristics include:

- Tools: Integrated Development Environments (IDEs), code editors, version control systems (e.g., Git).

- Flexibility: Developers can make changes and test them quickly.

- Debugging: Tools for debugging and profiling are often available.

- Isolation: Changes made in this environment do not affect other environments.

2. Testing Environment

The testing environment is where the application is tested for bugs and issues before it is deployed to production. Key characteristics include:

- Staging: Mimics the production environment to ensure accurate testing.

- Automated Testing: Tools for running automated tests (e.g., unit tests, integration tests).

- Quality Assurance: QA teams perform manual testing to identify issues.

- Data: Often uses a subset of production data to simulate real-world scenarios.

3. Production Environment

The production environment is where the application is live and accessible to users. Key characteristics include:

- Stability: Must be stable and reliable, as it directly impacts users.

- Performance: Optimized for performance and scalability.

- Monitoring: Tools for monitoring application performance and user activity.

- Backup and Recovery: Strategies in place for data backup and disaster recovery.

* Setting Up a Basic Environment in a Virtual Machine

Step 1: Choose a Virtualization Software

Select a virtualization software to create your virtual machine. Popular options include:

- VirtualBox: Free and open-source.

- VMware Workstation Player: Free for non-commercial use.

- Hyper-V: Built into Windows Pro and Enterprise editions.

Step 2: Download an Operating System

Choose an operating system to install on your VM. For this assignment, we will use Ubuntu, a popular Linux distribution.

1. Go to the [Ubuntu website](https://ubuntu.com/download).

2. Download the latest LTS (Long Term Support) version.

Step 3: Create a Virtual Machine

1. Open your virtualization software.

2. Create a new virtual machine:

- Allocate sufficient RAM (at least 2 GB).

- Allocate CPU cores (1-2 cores).

- Create a virtual hard disk (at least 20 GB).

3. Attach the downloaded Ubuntu ISO file to the VM.

Step 4: Install the Operating System

1. Start the virtual machine.

2. Follow the on-screen instructions to install Ubuntu.

3. Set up your user account and password.

Step 5: Set Up Development Tools

Once Ubuntu is installed, you can set up your development environment:

1. Install Git:

bash

sudo apt update

sudo apt install git

2. Install a Code Editor (e.g., Visual Studio Code):

bash

sudo snap install --classic code

3. Install Programming Languages (e.g., Python):

bash

sudo apt install python3 python3-pip

Step 6: Set Up a Testing Environment

To set up a basic testing environment, you can install testing frameworks:

1. Install pytest for Python:

bash

pip3 install pytest

2. Create a sample test file:

python

# test\_sample.py

def test\_addition():

assert 1 + 1 == 2

Step 7: Set Up a Production Environment

For a basic production environment, you can set up a web server:

1. Install Apache:

bash

sudo apt install apache2

2. Start the Apache service:

bash

sudo systemctl start apache2

3. Access the web server: Open a web browser and navigate to `http://localhost` to see the default Apache page.

**Q-8 Create a list of software you use regularly and classify them into the Following categories: system, application, and utility software.**

**Ans:-**

* System Software:
* Operating System:8
* Windows (or macOS, Linux, etc.) - The fundamental software that manages your computer's hardware and resources.
* Device Drivers:
* Software that allows your operating system to communicate with hardware components like printers, graphics cards, etc.
* Firmware:
* Embedded software within hardware devices like routers or BIOS (Basic Input/Output System) which is a type of firmware.
* Application Software:
* Web Browser: Chrome, Firefox, Safari, or Edge - Used for accessing and viewing websites.
* Email Client: Gmail, Outlook, or Thunderbird - For sending and receiving emails.
* Word Processor: Microsoft Word, Google Docs - For creating and editing text documents.
* Spreadsheet Software: Microsoft Excel, Google Sheets - For organizing and analyzing data in a tabular format.
* Presentation Software: Microsoft PowerPoint, Google Slides - For creating visual presentations.
* Media Player: VLC, Windows Media Player, iTunes - For playing audio and video files.
* Graphic Software: Adobe Photoshop, GIMP - For editing and creating images.
* Communication Software: Zoom, Slack, Microsoft Teams - For communication and collaboration.
* Games: Various video games downloaded or accessed online.
* Utility Software:
* Antivirus Software: Windows Defender, Norton, McAfee - Protects your computer from malware and viruses.
* Disk Cleanup: Built-in tool in Windows for removing unnecessary files.
* File Compression: 7-Zip, WinRAR - For compressing and decompressing files.
* Backup Software: Acronis True Image, Backblaze - For creating backups of your data.
* Disk Defragmenter: Built-in tool in Windows for optimizing disk performance.
* This list is not exhaustive but provides a good overview of the types of software and examples commonly used. The specific applications and utilities you use may vary based on your needs and preferences.

**Q-9 Follow a GIT tutorial to practice cloning, branching, and merging repositories**

**Ans:-**

1.Install Git

First, install Git on your computer from the official website.

2.Clone a Repository

Find a project on GitHub. Copy its link and use it to download the project files to your computer.

3.Open the Project Folder

Go into the folder where the project was downloaded.

4.Create a New Branch

Make a separate copy (branch) of the project to work on without changing the original.

5.Make Some Changes

Edit or add something to the project, like updating a file or writing new content.

6.Save and Record Your Changes

After editing, save the changes and tell Git to remember what you changed.

7.Go Back to the Main Version

Switch from your new branch back to the original (main) project version.

8.Merge Your Changes

Bring your changes from the new branch into the main version of the project.

9.Upload Changes Online

Send your final updated project back to GitHub so it’s saved online.

**Q-10 How does GIT improve collaboration in a software development team?**

**Ans-**

1.Keeps Everyone’s Work Organized

Git helps team members work on the same project without getting in each other’s way. Everyone can have their own copy to work on.

2.Tracks Every Change

Git keeps a full history of all changes made. If something breaks, the team can see who changed what and when.

3.Allows Working in Branches

Developers can create separate branches to work on different features or fixes. This means no one disturbs the main project while trying out new ideas.

4.Merges Changes Together Easily

After finishing work in a branch, changes can be combined (merged) into the main project. Git helps solve conflicts if two people worked on the same file.

5.Supports Teamwork from Anywhere

Since Git works with online platforms like GitHub or GitLab, team members can work together from different places.

6.Improves Communication

Git shows who made what changes and why. This helps the team understand each other’s work better.

7.Saves Time and Avoids Mistakes

Instead of sharing files manually, Git automatically manages versions. This saves time and avoids the risk of deleting or overwriting someone else's work.

**Q-11 Write a report on the various types of application software and how they improveproductivity.**

**Ans:-**

Introduction

Application software refers to programs designed to perform specific tasks for users. These are different from system software, which runs the computer itself. Application software helps people complete work more efficiently and accurately. In this report, we will explore different types of application software and understand how each one helps increase productivity in various fields.

* Types of Application Software:

1. Word Processing Software

Example: Microsoft Word, Google Docs

Use: Writing documents, reports, letters, resumes, etc.

Productivity Benefit: Allows easy editing, formatting, and spell-checking, saving time and effort compared to manual writing.

2. Spreadsheet Software

Example: Microsoft Excel, Google Sheets

Use: Data entry, calculations, financial analysis, charts

Productivity Benefit: Automates mathematical tasks, helps manage large amounts of data, and improves accuracy in calculations.

3. Presentation Software

Example: Microsoft PowerPoint, Canva, Google Slides

Use: Creating slideshows for meetings, teaching, or pitches

Productivity Benefit: Simplifies the process of sharing ideas visually and quickly during meetings or classes.

4. Database Management Software

Example: Microsoft Access, MySQL

Use: Storing and managing structured data

Productivity Benefit: Helps businesses manage information (like customer data) efficiently and retrieve it quickly when needed.

5. Email and Communication Software

Example: Microsoft Outlook, Gmail, Zoom, Slack

Use: Sending emails, video meetings, team messaging

Productivity Benefit: Speeds up communication, supports remote work, and reduces the need for in-person meetings.

6. Graphic Design and Multimedia Software

Example: Adobe Photoshop, CorelDRAW, Filmora

Use: Editing photos, creating visuals, making videos

Productivity Benefit: Enables professionals to create high-quality visuals faster, useful in marketing, media, and design.

7. Project Management Software

Example: Trello, Asana, Microsoft Project

Use: Organizing tasks, assigning work, tracking deadlines

Productivity Benefit: Keeps teams organized, helps track progress, and ensures projects are completed on time.

8. Web Browsers

Example: Google Chrome, Mozilla Firefox

Use: Accessing online tools, websites, and information

Productivity Benefit: Quick access to cloud apps, learning resources, and research material.

**Q-12 What is the role of application software in businesses?**

**Ans:-**

Application software plays a crucial role in businesses by enabling them to automate tasks, manage data, improve communication, and enhance overall productivity. It allows businesses to streamline operations, make more informed decisions, and ultimately achieve their goals.

Here's a more detailed look at the role of application software in businesses:

1. Automation and Efficiency:

• Application software automates repetitive tasks, reducing manual effort and minimizing errors.

• This leads to increased efficiency, allowing employees to focus on more strategic and value-added activities.

• Examples include automating invoice processing, inventory management, and payroll.

2. Data Management and Analysis:

• Businesses rely on application software to manage vast amounts of data, including customer information, financial records, and operational data.

• CRM (Customer Relationship Management) and ERP (Enterprise Resource Planning) systems are common examples.

• This data can then be analyzed to gain insights, improve decision-making, and identify areas for improvement.

3. Enhanced Communication and Collaboration:

• Application software facilitates communication and collaboration among employees, teams, and even with customers.

• Examples include email, instant messaging, video conferencing, and project management tools.

• This improved communication helps streamline workflows and fosters a more collaborative work environment.

4. Improved Customer Experience:

• Application software, particularly CRM systems, helps businesses manage customer interactions, personalize experiences, and improve customer satisfaction.

• By understanding customer needs and preferences, businesses can tailor their products and services to meet those needs, leading to increased customer loyalty and sales.

5. Strategic Decision-Making:

• By providing access to real-time data and analytical tools, application software empowers businesses to make more informed decisions.

• This data-driven approach allows businesses to identify trends, anticipate challenges, and adapt to changing market conditions.

6. Competitive Advantage:

• The efficient use of application software can give businesses a significant competitive advantage by improving productivity, reducing costs, and enhancing customer experience.

• By automating tasks, optimizing processes, and gaining valuable insights, businesses can outperform their competitors.

**Q-13 Perform a functional analysis for an online shopping system.**

**Ans:-**

1. Introduction:

An online shopping system allows users to browse, select, and purchase products over the internet. Functional analysis helps us understand what the system does and how users interact with it.

2. Functional Requirements (Main Features):

* User Registration and Login

Allows new users to sign up and existing users to log in Manages user accounts and stores customer details securely

* Product Browsing and Search

Lets users browse products by category (e.g., electronics, clothing)

Provides a search bar to find specific items using keywords or filters

* Product Details Page

Displays detailed information like product name, description, price, images, and reviews Includes an option to add the item to the cart

* Shopping Cart Management

Users can add, remove, or update product quantities in their cart Shows total cost before checkout

* Checkout Process

Collects shipping address and payment information Offers various payment methods (credit card, UPI, COD, etc.)Confirms the order and generates an invoice

* Order Management

Users can view their order history and current order status Admins can update order status (processing, shipped, delivered)

* User Reviews and Ratings

Allows customers to leave feedback on purchased products Helps other users make buying decisions

* Admin Panel (for Store Management)

Admins can add, update, or delete products Can manage users, view sales reports, and handle customer support

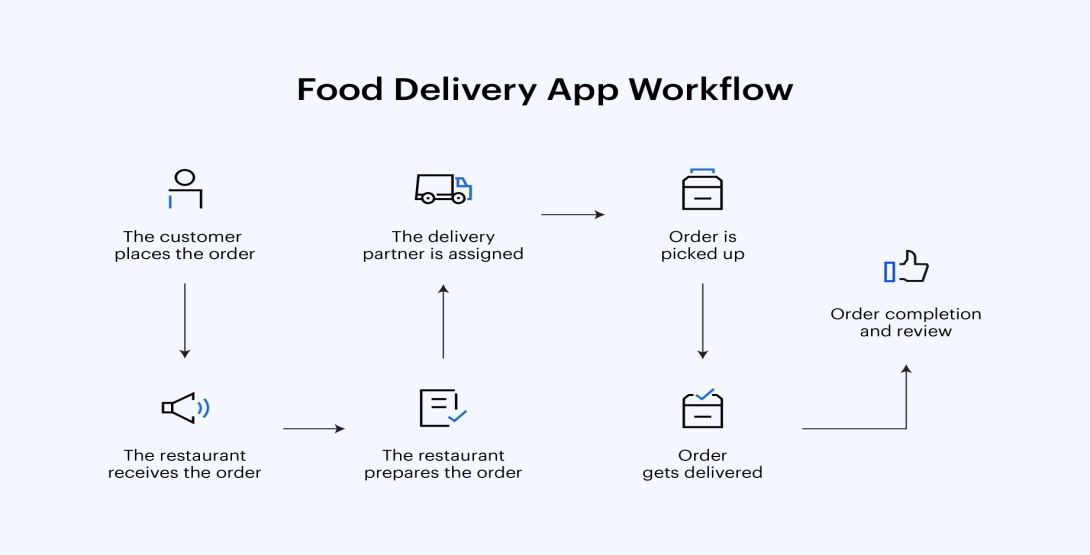
3. Additional Functionalities (Optional/Advanced):

* Wishlist Feature – Save products for later
* Discount Codes – Apply promotional offers
* Live Chat – Get instant support
* Notifications – Email or SMS alerts for order updates

**Q-14- Design a basic system architecture for a food delivery app**

**Ans:-**

.



A food delivery app connects customers, restaurants, and delivery agents through a structured system. It includes:

* Frontend (User Apps): Mobile or web apps for customers, restaurants, and delivery agents.
* Backend Server: Handles orders, user accounts, menus, and delivery logic.
* Database: Stores user data, orders, menus, and transaction history.
* External Services: Includes payment gateways, map APIs for tracking, and notification systems for real-time updates.

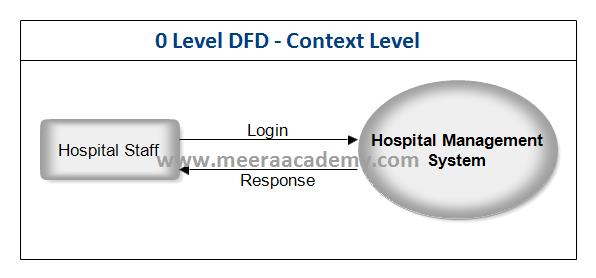
**Q-15- Document a real-world case where a software application required criticalmaintenance.**

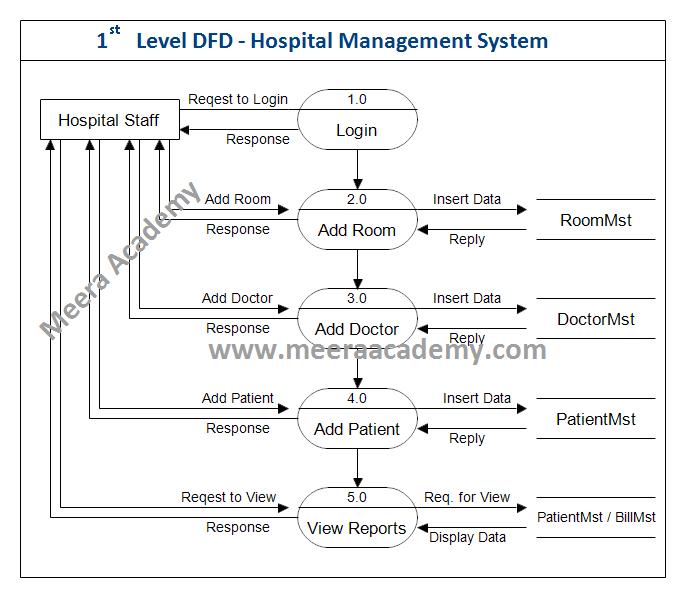
**Ans:-**

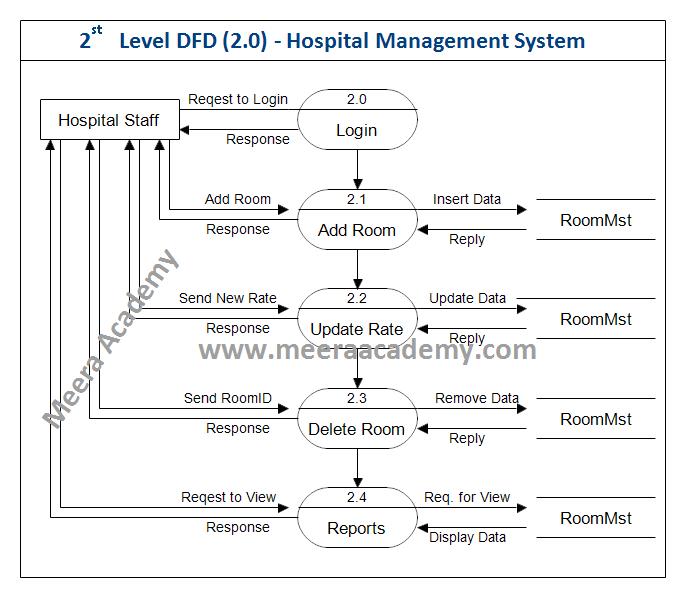
* The Problem:
* The Southwest Airlines reservation system experienced a major outage, rendering it unable to process bookings, check-ins, and other essential functions.
* This impacted a large number of flights and passengers, causing significant delays and cancellations across the airline's network.
* The outage exposed vulnerabilities in the system's architecture and maintenance procedures.
* The Cause:
* While the exact details of the outage were not publicly disclosed, it was widely reported that the issue stemmed from a software bug or configuration error.
* The bug likely triggered a cascade of problems, ultimately leading to the system-wide failure.
* This incident underscored the need for proactive maintenance, including thorough testing and monitoring of critical systems.
* The Maintenance Response:
* Southwest Airlines had to dedicate significant resources to identify and fix the issue, which involved a complex troubleshooting process.
* The airline implemented a series of emergency procedures to manage the immediate crisis, including manually checking in passengers and rebooking flights.
* The outage triggered a review of the airline's software maintenance practices and a commitment to improving the system's resilience.
* Lessons Learned:
* The Southwest Airlines outage serves as a cautionary tale about the critical role of software maintenance in ensuring the reliability and availability of essential services.
* It highlights the importance of regular system updates, thorough testing, and proactive monitoring to identify and address potential issues before they escalate.
* The incident also emphasizes the need for well-defined maintenance plans and skilled personnel to manage and maintain complex software systems.

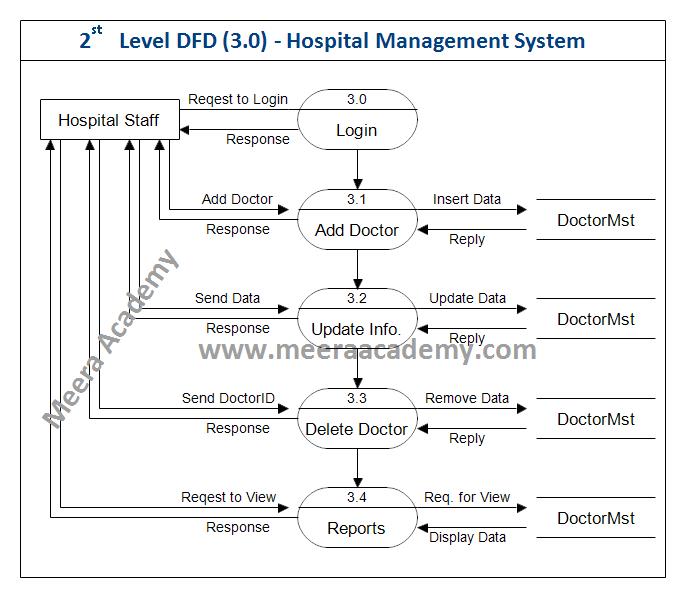
**Q-16- Create a DFD for a hospital management system.**

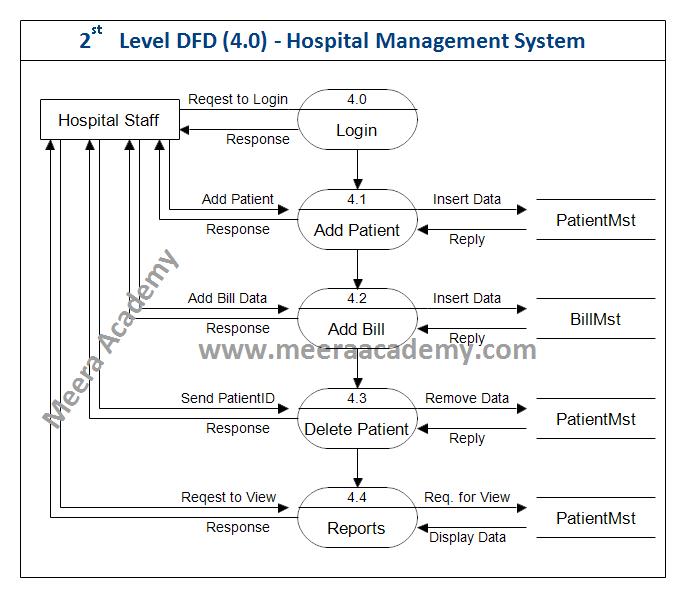
**Ans:-**

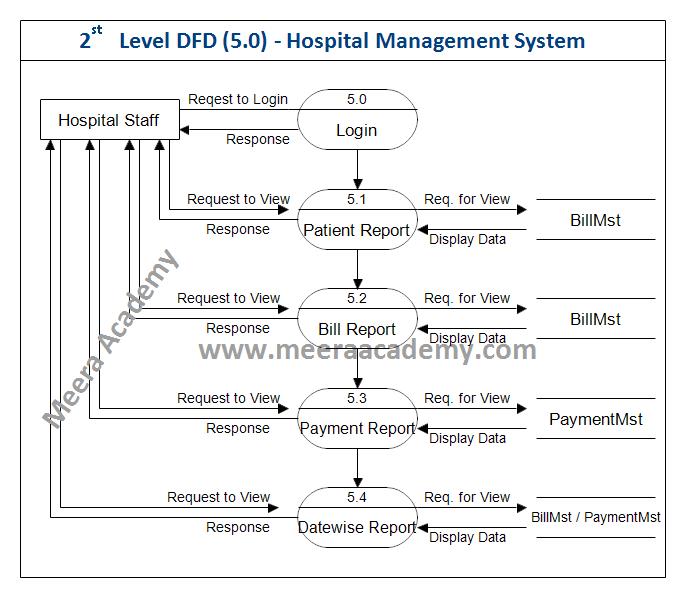


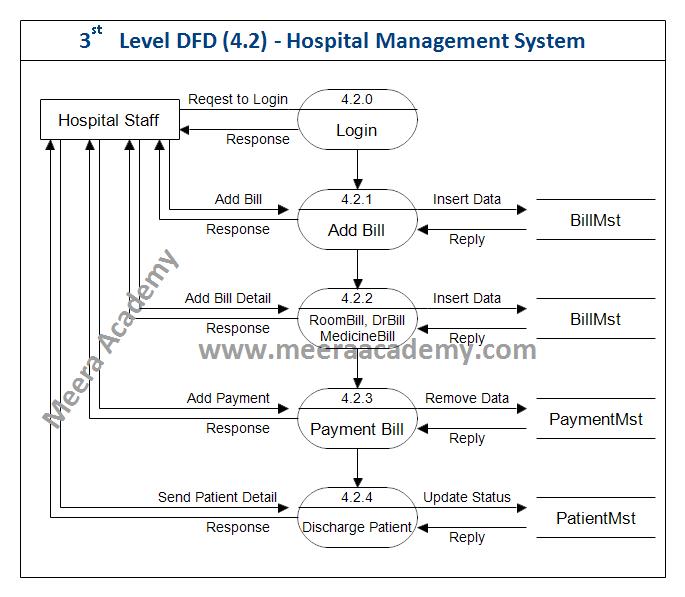






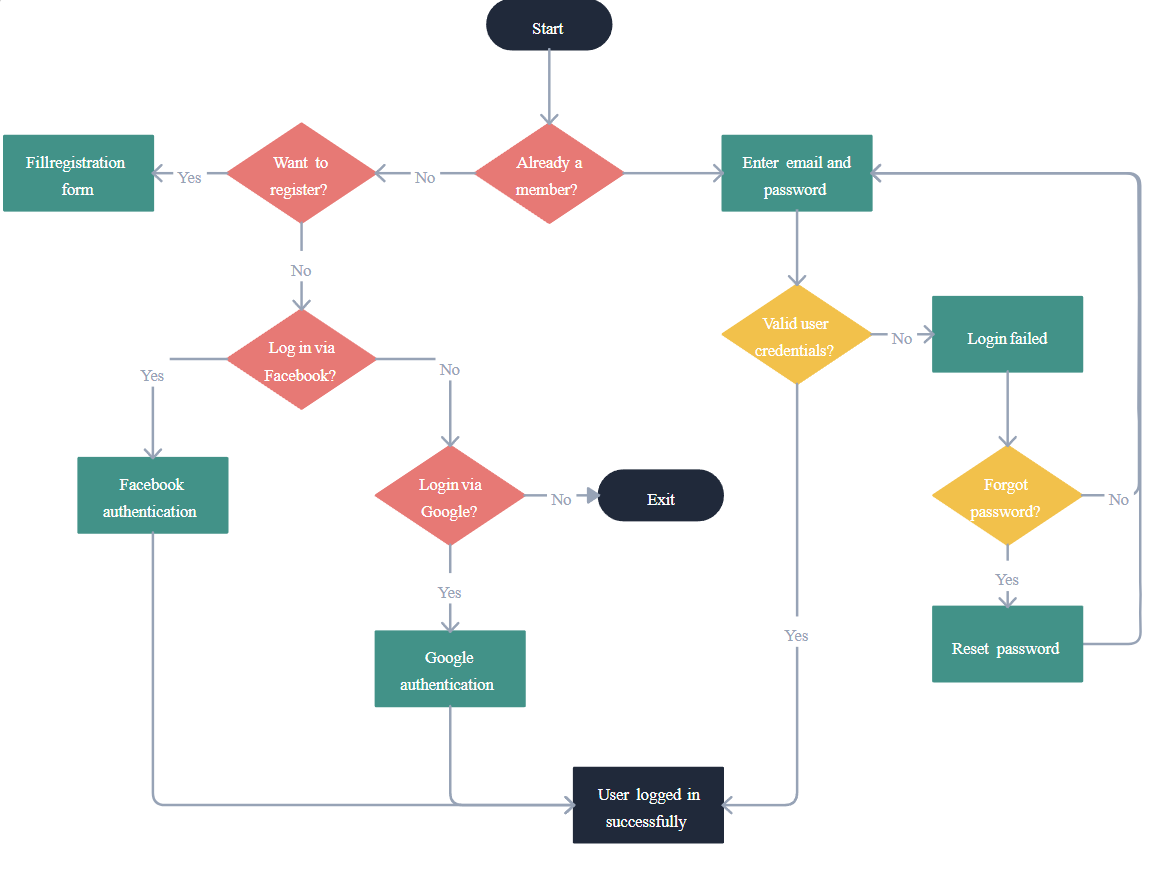






**Q-17 Draw a flowchart representing the logic of a basic online registration system.**

**Ans-**



**Q-18-What are the main stages of the software development process?**

**Ans:-**

1. Planning:

This initial phase involves defining the project's scope, objectives, and requirements. It includes feasibility studies, risk assessment, and resource planning.

2. Analysis:

This stage focuses on gathering and analyzing user needs to create detailed specifications for the software.

3. Design:

Here, the software architecture, user interface, and database design are created based on the requirements gathered in the previous phase.

4. Development (Coding):

This is where the actual coding of the software takes place, translating the design into functional code.

5. Testing:

This crucial phase involves various testing activities (unit testing, integration testing, system testing, user acceptance testing) to ensure the software meets quality standards and requirements.

6. Deployment:

This stage involves releasing the software to the users, which could involve installing it on servers, cloud platforms, or individual devices.

7. Maintenance:

After deployment, ongoing maintenance is required to address any bugs, improve performance, and add new features or functionality.

**Q-19-Write a requirement specification for a simple library management system.**

**Ans:-**

1. Introduction

The LMS will manage library resources, including books and members, facilitating book loans and returns.

2. Functional Requirements

* User Roles
* Librarian: Manages books, members, and transactions.
* Member: Searches for books and views loan status.
* Use Cases
* Book Management: Add, update, delete, and view books.
* Member Management: Register, update, delete, and view members.
* Loan Management: Loan and return books, view current loans.
* Member Features: Search books, view loan status, update account.

3. Non-Functional Requirements

* Performance

Support up to 100 concurrent users; search results within 2 seconds.

* Usability

Intuitive interface with help documentation.

* Security

User authentication and secure storage of sensitive information.

* Reliability

99.5% uptime and daily data backups.

* Compatibility

Works on major web browsers and mobile devices.

**Q-20-What are the key elements of system design?**

**Ans:-**

1-Architecture Design: Defines the overall structure of the system, including the components, their relationships, and how they interact. This can include client-server architecture, microservices, or monolithic designs.

2-Data Design: Involves defining the data structures, databases, and data flow within the system. This includes entity-relationship diagrams (ERDs), normalization of databases, and data storage solutions.

3-Interface Design: Focuses on how users will interact with the system. This includes designing user interfaces (UI), user experience (UX), and APIs for system integration. It ensures that the system is user-friendly and accessible.

4-Component Design: Breaks down the system into smaller, manageable components or modules. Each component should have a clear purpose and defined interfaces for interaction with other components.

5-Security Design: Addresses the security requirements of the system, including authentication, authorization, data encryption, and protection against vulnerabilities. This ensures that the system is secure from unauthorized access and data breaches.

6-Performance Design: Focuses on the system's performance requirements, including response time, throughput, and resource utilization. This involves optimizing algorithms, load balancing, and caching strategies.

7-Scalability Design: Ensures that the system can handle growth in users, data, and transactions without performance degradation. This may involve designing for horizontal or vertical scaling.

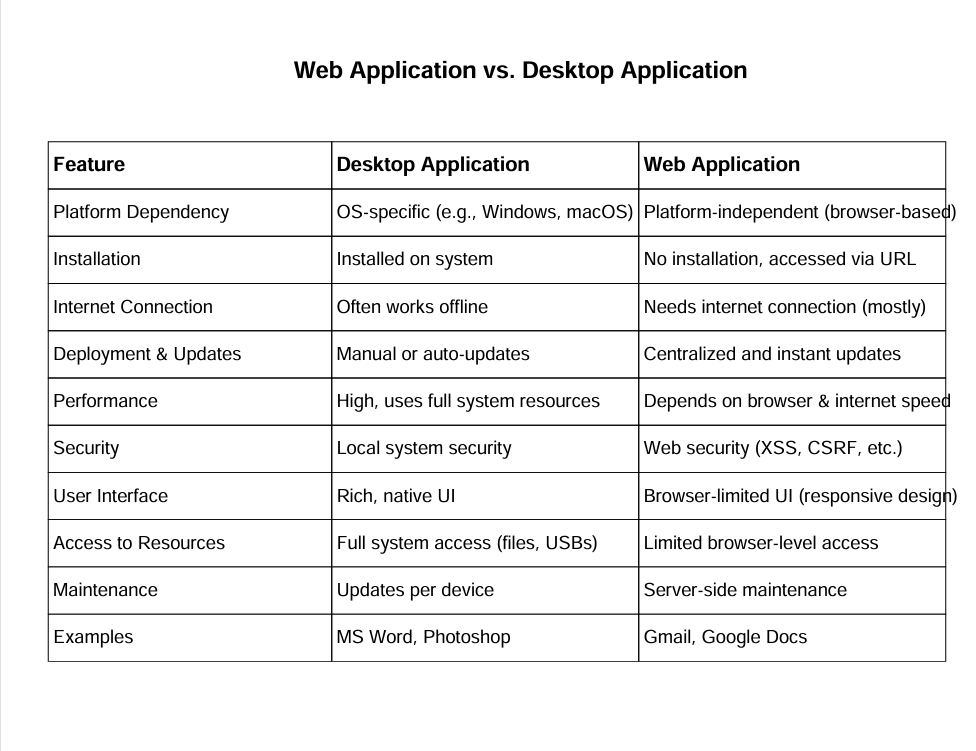
8-Maintainability Design: Considers how easy it will be to maintain and update the system over time. This includes code organization, documentation, and adherence to coding standards.

9-Deployment Design: Outlines how the system will be deployed in a production environment, including server configurations, cloud services, and continuous integration/continuous deployment (CI/CD) practices.

10-Testing Design: Defines the testing strategy, including unit testing, integration testing, system testing, and user acceptance testing (UAT). This ensures that the system meets its requirements and functions correctly.

**Q-21-What are the key differences between web and desktop applications?**

**Ans:-**

****

**Q-22- 5.Build a simple desktop calculator application using a GUI library.**

**Ans:-**

* Tools Used:
* Programming Language: Python
* GUI Library: Tkinter (built-in GUI library in Python)
* Platform: Windows or any OS with Python installed
* Project Description:

The calculator application allows the user to:

* Enter two numbers
* Choose an arithmetic operation
* See the result displayed on the screen
* Use buttons just like a real calculator

It features:

Number buttons (0–9)

Operation buttons (+, –, ×, ÷)

Clear (C) and Equal (=) buttons

Display area for input and output

Step-by-Step Process:

* Step 1: Design the Layout:-
* Use a simple window
* Add a display area (like a text field or label)
* Arrange buttons in a grid (numbers, operations, etc.)
* Step 2: Create the GUI Window:-
* Set the window title (e.g., "Simple Calculator")
* Define window size and background color (optional)
* Step 3: Add Buttons and Display:-
* Add number buttons (0–9)
* Add operator buttons (+, –, ×, ÷)
* Add "=" to calculate the result
* Add "C" to clear the screen
* Step 4: Write Functions:-
* A function to update the input when a button is clicked
* A function to evaluate the expression when "=" is pressed
* A function to clear the input when "C" is clicked
* Step 5: Run and Test :-
* Try different operations (e.g., 5 + 3 = 8)
* Check edge cases like division by zero
* Fix any errors or bugs

**Q-23 Design a simple HTTP client-server communication in any Language.**

**Ans:-**

1. Define the Server

Purpose: The server will listen for incoming HTTP requests and respond accordingly.

Protocol: Use the HTTP protocol, which operates over TCP/IP.

Endpoints: Define one or more endpoints (e.g., /, /hello) that the client can request.

Response Format: Decide on the response format (e.g., plain text, JSON).

Status Codes: Use standard HTTP status codes (e.g., 200 for success, 404 for not found).

2. Define the Client

Purpose: The client will send HTTP requests to the server and handle the responses.

Request Types: Typically, the client will use GET requests to retrieve data from the server.

Headers: Include necessary headers in the request (e.g., Content-Type, User -Agent).

Error Handling: Implement error handling for different HTTP status codes.

3. Communication Flow

Client Initialization: The client prepares to send a request.

Send Request: The client sends an HTTP request to the server's endpoint.

Server Receives Request: The server listens for incoming requests and processes them.

Server Response: The server generates a response based on the request and sends it back to the client.

Client Receives Response: The client receives the response and processes it (e.g., displaying the message or handling errors).

4. Example Scenario

Server: A simple HTTP server running on localhost at port 8080, responding to requests at the /hello endpoint with a message like "Hello, World!".

Client: A client application that sends a GET request to http://localhost:8080/hello and prints the server's response.

5. Tools and Technologies

Server: You can use various programming languages and frameworks (e.g., Python with Flask, Node.js with Express, Java with Spring).

Client: The client can be implemented in any language that supports HTTP requests (e.g., Python, JavaScript, Java).

6. Testing

Use tools like Postman or curl to manually test the server endpoints.

Ensure that the server responds correctly to valid requests and handles errors gracefully.