# Module 1: Overview of IT industry

# Lab Exercise

# Name: Vismay Thakker

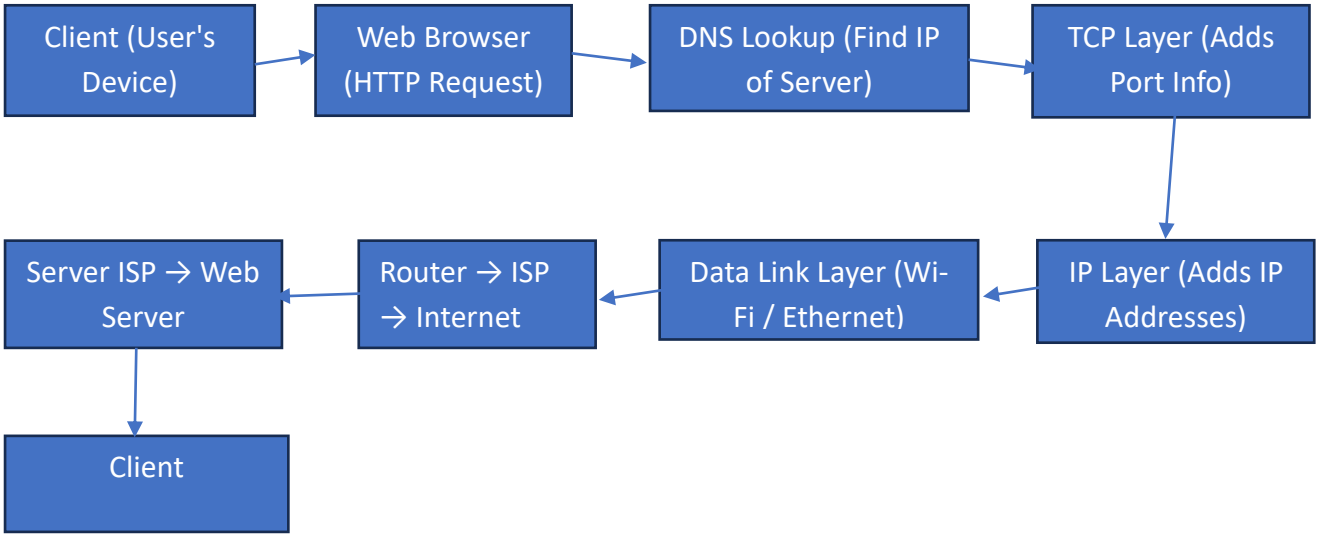
**1. Write a simple "Hello World" program in two different programming languages of your choice. Compare the structure and syntax.**

**Ans:** Done in c language and uploaded with name Q1.cpp and Q1.py

The "Hello World" program in C and Python highlights key differences in structure and syntax between the two languages. In C, the code is more structured and requires several elements such as including a header file (#include <stdio.h>), defining the main() function, and using braces {} to indicate code blocks. Statements must end with semicolons, and the program must be compiled before it can run. In contrast, Python is much simpler and more concise. It does not require any header files, and the print() function can be used directly without defining a main() function or using semicolons. Python relies on indentation to define code blocks, making it more readable and beginner-friendly. Overall, Python is easier and quicker for writing simple programs, while C gives more control over system-level operations but requires more boilerplate code.

**2. Research and create a diagram of how data is transmitted from a client to a server over the internet.**

**Ans:** Diagram:




When a client (such as a user's computer or smartphone) sends a request to a server (such as a website), the data travels through several steps and layers based on the OSI model or the TCP/IP model. Here's how it works:

- Client Request: The user enters a URL (like www.example.com) into a web browser.
- DNS Resolution: The browser contacts a DNS server to find the IP address of the server.
- HTTP Request: The browser prepares an HTTP request, which is part of the Application Layer.
- Transport Layer: The data is split into segments and assigned a port number using TCP (Transmission Control Protocol).
- Internet Layer: The data is packed into IP packets, with the client and server IP addresses added.
- Network Access Layer: The packets are converted into electrical/optical signals or radio waves and sent via Ethernet or Wi-Fi.
- Routers and Switches: The data passes through multiple routers and switches, which direct it towards the destination using the most efficient path.

- ISP and Internet Backbone: The data travels through the client's Internet Service Provider (ISP) and possibly across the global internet backbone.
- Server Side: The request reaches the server's ISP, then the target server, which processes the request (e.g., sends back a webpage).
- Response Back to Client: The server's response (also in packets) follows a similar path back to the client.

**3. Design a simple HTTP client-server communication in any language.**

**Ans:** Done

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

**Ans:**

1. Broadband (DSL and Cable)

Definition: Broadband includes DSL (Digital Subscriber Line) and cable internet, which are common in homes and businesses.

Pros:

-Widely available in urban and suburban areas

-Generally affordable

-Faster than dial-up connections

-Always-on connection (no need to dial in)

Cons:

-Slower than fiber

-Speed can vary depending on network traffic (especially cable)

-DSL speeds depend on the distance from the service provider's central location

2. Fiber Optic InternetDefinition: Uses fiber-optic cables to transmit data at extremely high speeds using light signals.

Pros:

-Extremely fast download and upload speeds

-Very low latency

-Reliable connection with minimal interference

-Great for streaming, gaming, and working from home

Cons:

-Limited availability (mostly in cities or high-demand areas)

-Installation can be expensive or unavailable in rural areas

3. Satellite Internet

Definition: Internet connection delivered via satellites orbiting Earth.

Pros:

-Available almost anywhere, especially rural or remote areas

-No need for physical cables

Cons:

-Slower speeds compared to broadband and fiber

-High latency (delay), which affects gaming and video calls

-Weather can impact connection quality

-Data caps may apply

4. Mobile Data (3G, 4G, 5G)

Definition: Internet access through cellular networks using SIM cards and mobile towers.

Pros:

-Portable — can be used on the go

-5G offers very fast speeds and low latency

-Widely available, especially in cities

Cons:

-Speeds and coverage depend on location and network congestion

-Data plans can be expensive or limited

-Less stable than wired connections

**5. Simulate HTTP and FTP requests using command line tools (e.g., curl).**

**Ans:**

Simulating HTTP Requests with curl

curl is a command-line tool used to transfer data using various network protocols including HTTP and FTP.

- Example 1: HTTP GET Request

curl http://example.com

This command fetches the homepage of example.com using an HTTP GET request.

- Example 2: HTTP POST Request

curl -X POST -d "username=admin&password=1234" http://example.com/login

This sends form data using an HTTP POST request to the /login endpoint.

- Example 3: Downloading a File via HTTP

curl -O http://example.com/file.zip

This downloads file.zip from the server and saves it with the same name.

Simulating FTP Requests with curl:

- Example 1: FTP File Listing

curl ftp://ftp.example.com/ --user username:password

Lists files on the FTP server after logging in with the given credentials.

- Example 2: FTP File Download

curl -u username:password ftp://ftp.example.com/file.txt -O

Downloads file.txt from the FTP server using FTP credentials.

- Example 3: FTP File Upload

curl -T localfile.txt ftp://ftp.example.com/upload/ --user username:password

Uploads localfile.txt to the /upload/ directory on the FTP server.

**6. Identify and explain three common application security vulnerabilities. Suggest possible solutions.**

**Ans:** Three Common Application Security Vulnerabilities and Suggested Solutions

1. SQL Injection
Description: SQL Injection is a vulnerability that occurs when an

attacker is able to insert or manipulate SQL queries through input fields. This can lead to unauthorized access to, or manipulation of, the database.
Solution: To prevent SQL injection, developers should use parameterized queries or prepared statements, which safely separate user input from SQL commands. Additionally, input validation and sanitization should be applied consistently.

2. Cross-Site Scripting (XSS)
Description: Cross-Site Scripting allows attackers to inject malicious scripts into web pages viewed by other users. These scripts can steal sensitive data such as cookies or session tokens.
Solution: Developers should validate and encode all user inputs before displaying them in a web browser. Implementing Content Security Policy (CSP) headers can also help restrict the execution of untrusted scripts.

3. Broken Authentication
Description: This vulnerability arises when authentication mechanisms are poorly implemented, allowing attackers to compromise user accounts.
Solution: Secure authentication can be ensured by enforcing strong password policies, implementing multi-factor authentication (MFA), and managing session tokens securely—such as using secure cookies and automatic session timeouts.

**7. Identify and classify 5 applications you use daily as either system software or application software.**

**Ans:** Five Applications I Use Daily and Their Classification

1. Google Chrome – *Application Software*
Used for browsing the internet, watching videos, and accessing web-based tools.

2. Microsoft Word – *Application Software*
Used for creating and editing documents, reports, and assignments.

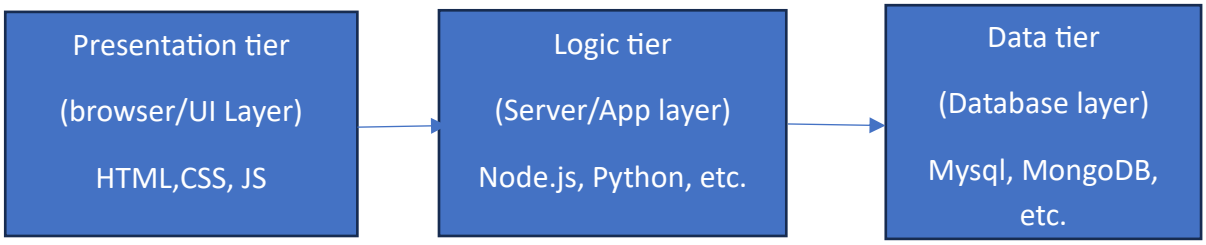
3. Windows Operating System – *System Software*
Manages the computer hardware and provides a platform for running applications.

4. Spotify – *Application Software*
Used to stream and listen to music or podcasts.

5. Antivirus Software (e.g., Windows Defender) – *System Software*
Protects the computer from malware, viruses, and other threats.

**8. Design a basic three-tier software architecture diagram for a web application.**

**Ans:**




A three-tier architecture divides a web application into three main layers: Presentation Tier, Logic Tier, and Data Tier. Each layer has a specific role in handling the application's functionality.

1. Presentation Tier (Client Layer)

This is the front-end or user interface.
It includes web pages and forms users interact with.
Technologies used: HTML, CSS, JavaScript, front-end frameworks (e.g., React, Angular).

2. Logic Tier (Application Layer)

This is the back-end where the business logic is processed.
It handles requests from the user interface and communicates with the database.
Technologies used: Node.js, Python (Django/Flask), Java (Spring), etc.

3. Data Tier (Database Layer)

This layer stores, retrieves, and manages data.
It includes the database management system (DBMS).
Technologies used: MySQL, PostgreSQL, MongoDB, etc.

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

**Ans:**

Case Study: Functionality of the Three Layers in an Online Food Ordering System
Software System Example: Online Food Ordering Application (like zomato or swiggy)
This system allows users to browse restaurants, place orders, and make payments through a web or mobile app. It is built using a three-tier architecture, which includes the Presentation Layer, Business Logic Layer, and Data Access Layer.

**1. Presentation Layer (User Interface Layer)**

Function: This is the layer that users interact with directly.

How it works in this case:
-Displays restaurant lists, food menus, and order details.
-Allows users to search for food, select items, and place orders.
-Technologies used: HTML, CSS, JavaScript, mobile UI frameworks.

Example: A user opens the app, browses nearby restaurants, and adds food items to their cart.

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

Function: This layer processes user requests and applies the rules of the system.

How it works in this case:

-Calculates total cost, delivery fees, and taxes.

-Verifies if items are in stock before confirming the order.

-Sends order details to the correct restaurant and delivery personnel.

-Handles payment processing and user authentication.

Example: When a user clicks "Place Order," this layer checks if the selected food is available, calculates the final amount, and initiates the payment process.

**3. Data Access Layer (Database Layer)**

Function: This layer manages all interactions with the database.
How it works in this case:
Retrieves restaurant and menu data.
Stores order details, user information, and transaction history.
Updates order status and stock availability in real-time.
Example: After an order is placed, this layer saves the order in the database and updates the item availability.

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

**Ans:** In software development, environments are isolated spaces used to develop, test, and deploy applications. Each environment serves a

specific purpose and ensures the software is functioning correctly at every stage before reaching the end user.

1. Development Environment
-Purpose: For developers to write and test code.
-Characteristics:
>Includes code editors (e.g., VS Code), compilers, interpreters.
>Often has debugging tools.
>May have mock databases or local versions of services.
>Example Tools: Git, Docker, Node.js, local databases like SQLite.

2. Testing (Staging) Environment
Purpose: To test the software in a controlled setting similar to production.
Characteristics:
Mirrors production setup as closely as possible.
Used for quality assurance (QA), integration testing, and user acceptance testing (UAT).
May contain anonymized or mock data.
Example Tools: Jenkins (CI/CD), Selenium, Postman, TestRail.

3. Production Environment
Purpose: Where the application runs and is accessed by real users.
Characteristics:
Live and stable environment.
Requires high reliability and performance.
Has real data and is often monitored for uptime and performance.
Example Tools: Kubernetes, Nginx, AWS, Datadog.

**11. Write and upload your first source code file to Github.**

**Ans:** Done

**12. Create a Github repository and document how to commit and push code changes.**

Ans: Step 1: Create a GitHub Repository

-Go to https://github.com and log in to your account.

-Click the + icon in the top right corner and select "New repository".

-Fill in the repository name (e.g., my-first-repo).

-Add a description (optional), choose Public or Private, and check "Initialize this repository with a README" if desired.

-Click "Create repository".

Step 2: Clone the Repository to Your Local Machine

```
git clone https://github.com/your-username/my-first-repo.git
cd my-first-repo
```

Step 3: Make Changes to Your Code

-Open the repository folder in a code editor like VS Code.

-Create or modify any file, for example:

```
echo "Hello, GitHub!" > hello.txt
```

Step 4: Commit and Push Changes

a. Stage the changes

```
git add .
```

This stages all modified files for commit.

b. Commit the changes with a message

```
git commit -m "Add hello.txt with greeting"
```

This records the changes in your local Git history.

c. Push the changes to GitHub

git push origin main

Step 5: Verify on GitHub

-Go to your repository page on GitHub.

-Refresh and confirm that the new file and commit message appear.

**13. Create a student account on Github and collaborate on a small project with a classmate.**

**Ans:** Done

**14. Create a list of software you use regularly and classify them into the following categories: system, application, and utility software.**

**Ans:**

1. System Software:
Windows 11
Ubuntu
Device Drivers
BIOS / UEFI

2. Application Software:
Microsoft Word
Google Chrome
Zoom
Spotify
Visual Studio Code
Microsoft Excel

3. Utility Software:
Windows Defender
CCleaner

WinRAR / 7-Zip
Disk Management Tool
Backup & Restore

**15. Follow a GIT tutorial to practice cloning, branching, and merging repositories.**

**Ans:** Done

**16. Write a report on the various types of application software and how they improve productivity.**

**Ans:** Application software refers to programs designed to help users perform specific tasks. These tasks can range from writing documents to managing data or communicating online. Such software plays a key role in improving efficiency and productivity in both personal and professional settings.
Types of Application Software

1.Word Processing Software
*Examples: Microsoft Word, Google Docs*
Helps create and edit documents quickly and professionally.

2.Spreadsheet Software
*Examples: Excel, Google Sheets*
Automates calculations and organizes data, saving time and effort.

3.Presentation Software
*Examples: PowerPoint, Google Slides*
Allows users to present ideas clearly using slides, images, and charts.

4.Communication Software
*Examples: Zoom, Microsoft Teams*

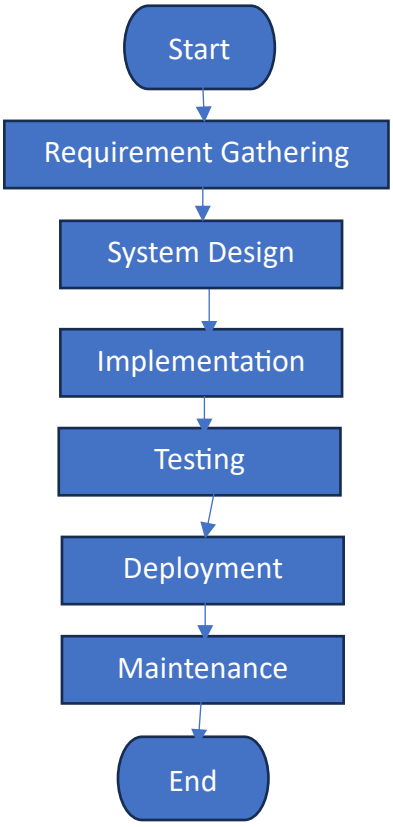
Enhances teamwork and remote communication through video calls and messaging.

5.Multimedia Software
*Examples: VLC Media Player, Adobe Photoshop*
Supports creating and editing audio, video, and graphics for personal or professional use.

6.Web Browsers
*Examples: Google Chrome, Firefox*
Provides access to the internet, online tools, and cloud-based applications.

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

**Ans:**



**18. Write a requirement specification for a simple library management system.**

**Ans:**

1. Introduction

This document defines the functional and non-functional requirements for a simple Library Management System. The system will help manage books, users, and borrowing activities in a library.

2. Functional Requirements

2.1 User Management

-The system shall allow the registration of new users (students, staff).
-The system shall allow login and logout functionality.
-The system shall assign roles (admin or user).

2.2 Book Management

-The system shall allow the admin to add, update, or delete book records.
-The system shall allow users to search for books by title, author, or ISBN.
-The system shall display book availability status (available/borrowed).

2.3 Borrowing and Returning Books

-The system shall allow users to borrow available books.
-The system shall track due dates for borrowed books.
-The system shall allow users to return books.
-The system shall update the book status after return.

2.4 Fine Management

-The system shall calculate overdue fines based on the number of late days.
-The system shall notify users of any unpaid fines.
-Reports and Logs
-The system shall generate reports of issued books, returned books,

and overdue books.
-The system shall keep logs of user activity (borrowing/returning history).

3. Non-Functional Requirements

-Usability: The system should have a user-friendly interface.
-Performance: The system should respond to user actions within 2 seconds.
-Security: Only authorized users can access or modify data.
-Reliability: The system should ensure data is saved and backed up regularly.
-Scalability: The system should support a growing number of books and users.

4. System Users
Admin: Manages books, users, and system settings.
User (Student/Staff): Searches books, borrows and returns books, views borrowing history.

**5. Assumptions and Constraints**
The system is web-based and accessible through a browser.
Only one library branch is managed.
No online payment is included for fine management.

**19. Perform a functional analysis for an online shopping system.**

**Ans:**
Functional Analysis for an Online Shopping System

1. User Registration and Authentication
-Functionality: Users can create an account, log in, and reset passwords. Social media logins may be supported.
-Goal: Secure and personalized user access.

2. Product Search and Browsing
Functionality: Search products by keywords, categories, or filters. Sort by price, ratings, or popularity.
Goal: Easy product discovery.

3. Shopping Cart Management
Functionality: Users can add, remove, and edit items in the cart. Display total price, taxes, and shipping.
Goal: Review and modify purchases before checkout.

4. Checkout and Payment Processing
Functionality: Users enter shipping details and payment info to complete their purchase. Support for promo codes.
Goal: Secure and efficient order processing.

5. Order Management
Functionality: Track orders, view status updates, and manage returns or cancellations.
Goal: Transparency and control over the order process.

6. Customer Reviews and Ratings
Functionality: Users can rate and review products, and filter reviews.
Goal: Help customers make informed decisions.

7. Security Features
Functionality: Use encryption, secure login, and payment processing.
Goal: Protect user data and transactions.

8. Admin Panel
Functionality: Admins can manage users, products, orders, and reports.
Goal: Efficient system management.

**20. Design a basic system architecture for a food delivery app.**

**Ans:** System Architecture for a Food Delivery App

1. Client-Side (Mobile App/Web App)
Purpose: The client-side is where users (customers and drivers) interact with the app.
Components:
-User Interface (UI): Designed for customers and drivers. The customer UI allows browsing restaurants, placing orders, making payments, etc. The driver UI allows them to view and accept delivery orders.
-User Authentication: Allows customers and drivers to register/login using email, social media accounts, or phone numbers.
Push Notifications: Used for real-time alerts (order updates, promotions, etc.).

2. API Gateway
Purpose: Acts as a bridge between the client-side app and the server-side components, handling requests and responses.
Components:
-Request Routing: Routes incoming requests from the client to the appropriate backend service (e.g., orders, payment, restaurant data).
Authentication and Authorization: Ensures users are authenticated before accessing certain services (e.g., placing an order).

3. Backend Services
Purpose: Responsible for handling business logic, processing requests, and managing data.
Components:
-User Service: Manages customer and driver accounts, authentication, and profile data.
-Restaurant Service: Handles restaurant data, menus, and availability.
-Order Service: Manages order placement, updates, and status tracking.

-Payment Service: Processes payments via integrated payment gateways (e.g., Stripe, PayPal).
Notification Service: Sends real-time notifications (order updates, delivery status).
-Rating and Review Service: Allows customers to rate restaurants and drivers.

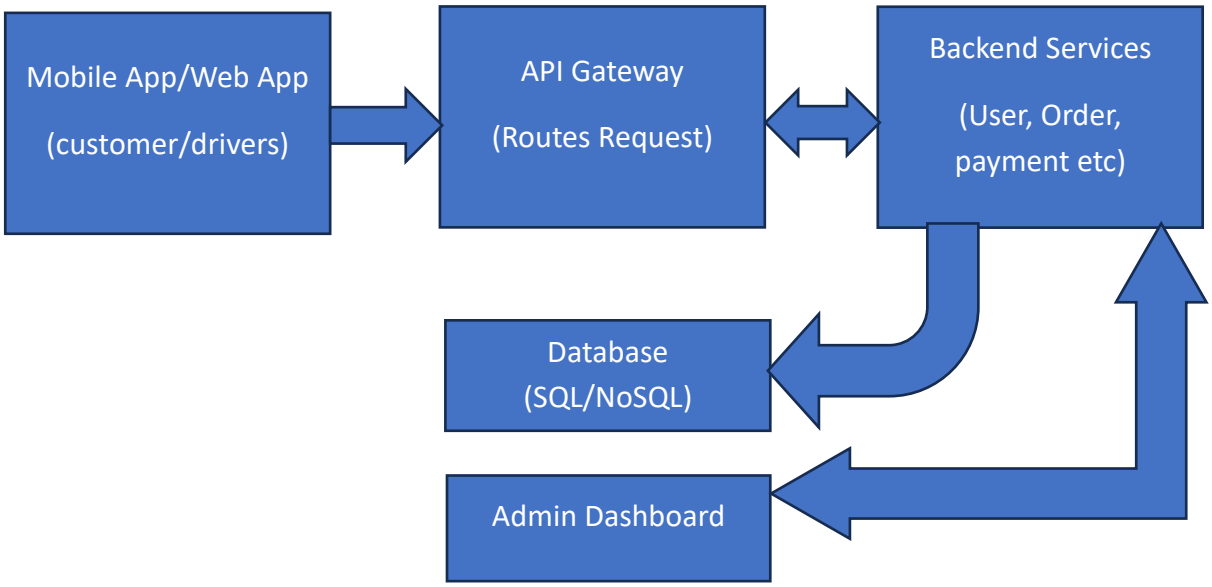
4. Database
Purpose: Stores all system data, including user profiles, orders, restaurant details, and payment information.
Types of Databases:
Relational Database (e.g., PostgreSQL/MySQL): Stores structured data such as user profiles, order details, restaurant menus, etc.
NoSQL Database (e.g., MongoDB): Stores unstructured or semi-structured data like user reviews, ratings, and logs.

5. Admin Dashboard
Purpose: Provides the backend admin interface for managing users, restaurants, orders, and analytics.
|Components:
-Restaurant Management: Admin can manage restaurant listings, menus, and availability.
-Order Management: Admin can view and manage orders, cancellations, and refunds.
-Analytics: Provides insights on sales, order trends, and user behavior.

6. Payment Gateway Integration
Purpose: Facilitates secure transactions for order payments.
Components:
-Payment Processor: Integrates with payment gateways like Stripe, PayPal, or credit card systems to handle payments securely.
-Transaction Records: Ensures payment transactions are logged and tracked for reporting.

7. Geolocation and Mapping Service
Purpose: Provides location-based services for order tracking and delivery routing.
Components:
-GPS Integration: Tracks customer and driver locations in real-time.
-Mapping API (e.g., Google Maps): Provides navigation for drivers and calculates estimated delivery time.

8. Security
Purpose: Protects user data and ensures secure transactions.
Components:
-SSL Encryption: Secures communication between the client and server.
-OAuth2 or JWT (JSON Web Tokens): Secure authentication methods for users.
-Data Privacy: Ensures sensitive data like payment info is encrypted.



**21. Develop test cases for a simple calculator program.**

**Ans:**

Test Case 1: Addition
-Test Description: Test the addition of two positive integers.
-Input: 5 + 3
-Expected Output: 8
-Pass/Fail Criteria: The program should return the sum of 5 and 3, which is 8.

Test Case 2: Subtraction
-Test Description: Test the subtraction of a smaller number from a larger number.
-Input: 10 – 4
-Expected Output: 6
-Pass/Fail Criteria: The program should return the result of 10 minus 4, which is 6.

Test Case 3: Multiplication
-Test Description: Test the multiplication of two numbers.
-Input: 6 * 7
-Expected Output: 42
-Pass/Fail Criteria: The program should return the product of 6 and 7, which is 42.

Test Case 4: Division (Valid Input)
-Test Description: Test the division of two numbers where the denominator is not zero.
-Input: 20 / 4
-Expected Output: 5
-Pass/Fail Criteria: The program should return the result of 20 divided by 4, which is 5.

Test Case 5: Division by Zero
-Test Description: Test division by zero (an invalid operation).
-Input: 10 / 0
-Expected Output: Error: Division by zero
-Pass/Fail Criteria: The program should return an error message indicating that division by zero is not allowed.

Test Case 6: Negative Number Addition
-Test Description: Test the addition of a positive and a negative number.
-Input: 5 + (-3)
-Expected Output: 2
-Pass/Fail Criteria: The program should return the result of 5 plus -3, which is 2.

Test Case 7: Decimal Number Addition
-Test Description: Test the addition of two decimal numbers.
-Input: 5.5 + 3.2
-Expected Output: 8.7
-Pass/Fail Criteria: The program should return the sum of 5.5 and 3.2, which is 8.7.

Test Case 8: Multiplying with Zero
-Test Description: Test the multiplication of a number with zero.
-Input: 7 * 0
-Expected Output: 0
-Pass/Fail Criteria: The program should return 0 since any number multiplied by zero equals zero.

Test Case 9: Subtraction Resulting in Negative Number
-Test Description: Test subtraction where the result is a negative number.
-Input: 3 – 8
-Expected Output: -5
-Pass/Fail Criteria: The program should return the result of 3 minus 8, which is -5.

Test Case 10: Large Number Multiplication
-Test Description: Test multiplication of large numbers.
-Input: 10000 * 2500
-Expected Output: 25000000
-Pass/Fail Criteria: The program should correctly compute large number multiplication.

Test Case 11: Input with Invalid Characters
-Test Description: Test the calculator with invalid input (e.g., non-numeric characters).
-Input: a + b
-Expected Output: Error: Invalid input
-Pass/Fail Criteria: The program should return an error message indicating invalid input.

Test Case 12: Parentheses in Operations (if supported)
-Test Description: Test the use of parentheses in arithmetic operations.
-Input: (5 + 3) * 2
-Expected Output: 16
-Pass/Fail Criteria: The program should correctly evaluate the

expression inside the parentheses first, then perform the multiplication.

Edge Case Test Cases

Test Case 13: Floating Point Precision
-Test Description: Test operations that may involve floating-point precision.
-Input: 0.1 + 0.2
-Expected Output: 0.3
-Pass/Fail Criteria: The program should correctly add floating-point numbers, though care must be taken to account for potential floating-point precision issues.

Test Case 14: Large Result
-Test Description: Test large number addition.
-Input: 99999999 + 99999999
-Expected Output: 199999998
-Pass/Fail Criteria: The program should handle large results and return the correct sum.

Test Case 15: Multiple Operations
-Test Description: Test a sequence of operations in one expression.
Input: 10 + 2 * 3
-Expected Output: 16
-Pass/Fail Criteria: The program should follow the correct order of operations (PEMDAS: Parentheses, Exponents, Multiplication/Division, Addition/Subtraction).

**22. Document a real-world case where a software application required critical maintenance.**

**Ans:**

Real-World Case: Critical Maintenance in WhatsApp (2021 Outage)

Overview:
On October 4, 2021, WhatsApp, along with Facebook and Instagram (all owned by Meta), experienced a global outage lasting about 6 hours. This was one of the most significant outages for these platforms, affecting billions of users worldwide.
Cause of the Issue:
-The outage was caused by a misconfiguration during routine maintenance of Meta's backbone routers, which manage traffic between data centers.
-This misconfiguration unintentionally disconnected Facebook's data centers from the internet.
-The problem was worsened because internal tools used by engineers to fix the issue were also unreachable due to the same network breakdown.

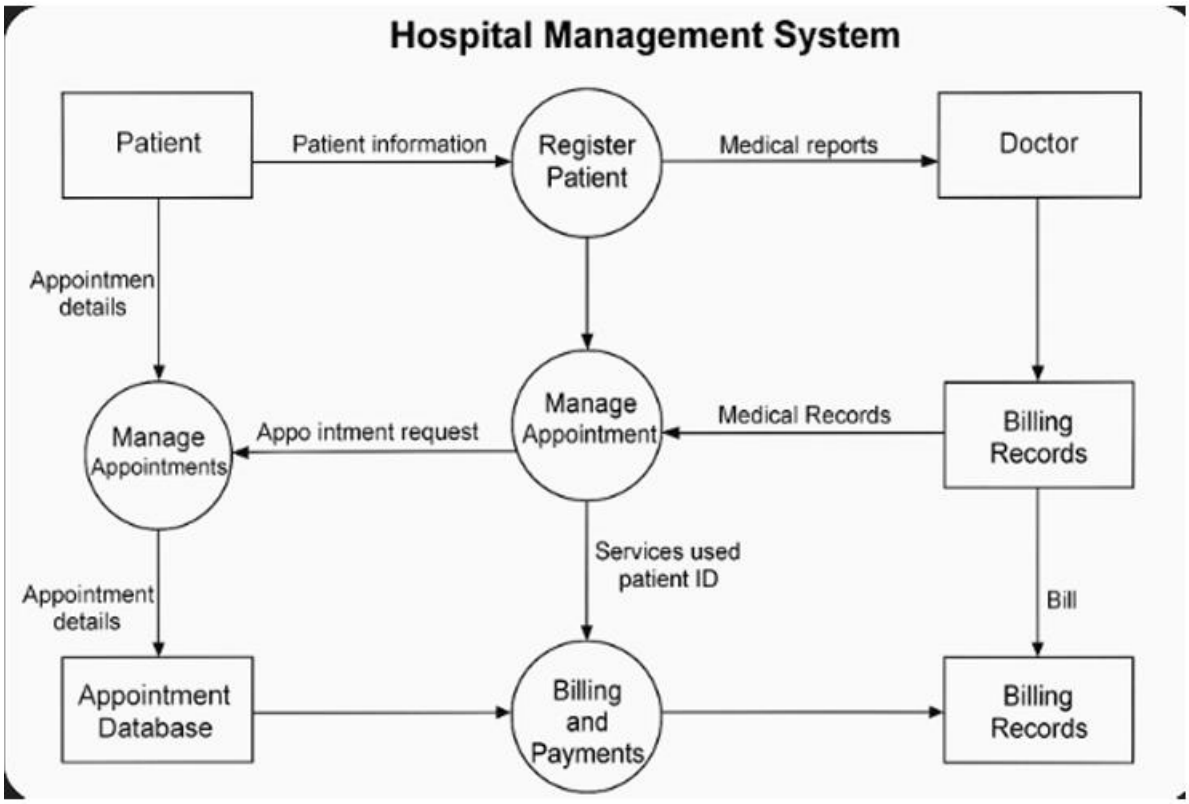
Critical Maintenance Actions Taken:
-Engineers had to physically access data center servers to manually reset systems.
-Network configurations were reviewed and updated to prevent the same type of misconfiguration from occurring again.
-Meta conducted a detailed post-mortem and implemented better internal tooling and rollback systems for future updates.

Lessons Learned:
-Always test configuration changes in isolated environments before applying them to production systems.
-Implement strong fallback and recovery mechanisms.

-Ensure internal tools and systems have separate access channels in case of outages.

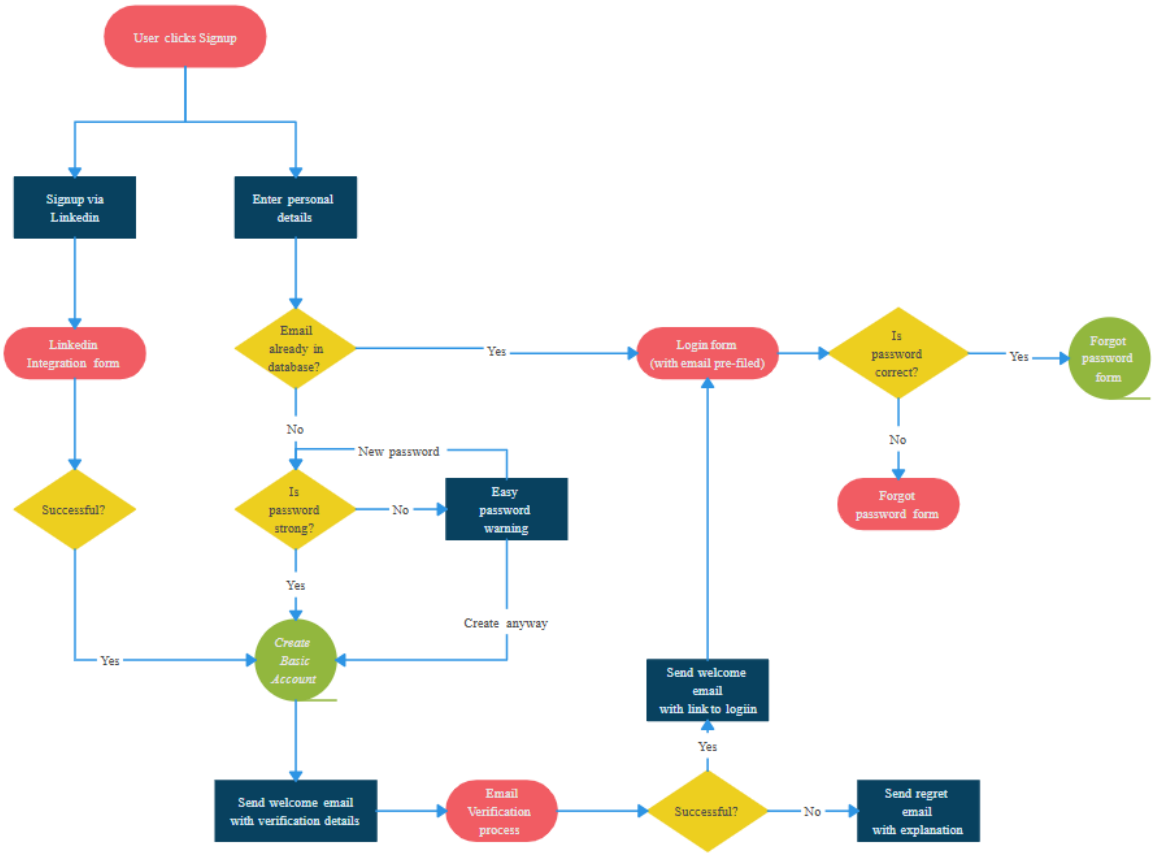
**23. Create a DFD for a hospital management system.**

**Ans:**




**24. Build a simple desktop calculator application using a GUI library.**

**Ans:** Done and uploaded to github with name Q24.py

**25. Draw a flowchart representing the logic of a basic online registration system.**

**Ans:**