Assignment – 1

Module 1 - OVERVIEW OF IT INDUSTRY

1 ) What is a Program?

Program is set of instuction that perform the specific task .

2)LAB EXERCISE:

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

C Programming Language

#include<stdio.h>

int main ()

{

printf("Hello world ");

return 0;

}

Python Programming Language

print("Hello, World!")

Comparision of both Language

1. C is a low-level language, which is faster and closer to machine code, while Python is a high-level language that is easier to read and write.

2. C needs manual memory management, but Python handles memory automatically.

3. Python is better for beginners due to its simple syntax, while C requires more knowledge of how the computer works.

3) THEORY EXERCISE:

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 works like a recipe, where each step is followed in order to complete a specific task, like adding numbers, displaying a message, or saving a file.

When you run a program, the computer reads the instructions line by line, understands them (through a compiler or interpreter), and performs the actions accordingly. The program can take input from the user, process it, and give output based on the logic written by the programmer.

In short, a program helps us control and communicate with the computer to solve problems or perform tasks automatically.

2 . What is Programming?

Programming is the process of writing instructions (called code) that a computer can understand and follow to perform specific tasks .

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

Follwing are the step involved in programming Process

1. Understand the Problem

Know what you want the program to do.

2. Plan the Solution

Think and write down the steps to solve the problem (like making a flowchart or algorithm).

3. Write the Code

Use a programming language (like Python or C) to write the instructions.

4. Test the Program

Run the program to check if it works correctly or has any errors (bugs).

5. Fix Errors (Debugging)

If there are mistakes, correct them and test again.

6. Final Output

Once it works perfectly, your program is ready to use.

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

| Feature | High-Level Language | Low-Level Language |
| --- | --- | --- |
| Ease of Use | Easy to read, write, and understand | Hard to understand, closer to machine language |
| Examples | Python, Java, C++, JavaScript | Assembly language, Machine code |
| Closer To | Human language | Computer hardware |
| Portability | Works on different computers with little change | Depends on specific hardware |
| Memory Management | Mostly automatic | Manual, programmer must manage it |
| Speed | Slower compared to low-level | Faster, more efficient |

5. World Wide Web & How Internet Works ?

The World Wide Web (WWW) is a system of web pages and websites stored on the internet. It uses browsers (like Chrome or Firefox) to access information. The web works through hyperlinks, allowing users to click and jump from one page to another.

* It was invented by Tim Berners-Lee in 1989.
* Uses HTTP/HTTPS to transfer web data.
* Web pages are written using HTML (HyperText Markup Language).

How internet Works ?

1. You type a website address (like www.google.com) into your browser.
2. Your computer sends a request to a **DNS (Domain Name System)** to find the server's IP address.
3. The request goes through internet networks to reach the website’s web server.
4. The server sends back the web page data to your browser.
5. Your browser displays the website on your screen.

4 ) Lab Exercies : Research and create a diagram of how data is transmitted from a client to a server over the internet.

REQUEST

CLINTT

PHNE-PT

SERVER

NETWORK

PC-PT

LAPTOP-PT

RESPONCE

Steps in Data Transmission (Client to Server)

1. User Action:  
 You click on something – like opening a website or sending a message.

2. Request Created:  
 Your device creates a request (like "Hey server, show me this website").

3. Data is Broken into Packets:  
 The request is split into small pieces called packets.

4. Sent Through Your Router:  
 The packets go to your Wi-Fi router or modem.

5. Goes to Internet Provider (ISP):  
 From the router, the data goes to your Internet Service Provider (like Jio, Airtel, etc.).

6. Travels Through the Internet:  
 The data travels across many routers and networks until it reaches the right server (the computer that stores the website or app).

7. Server Gets the Request:  
 The server reads the request, finds the data (like a webpage), and sends it back.

8. Data Comes Back to You:  
 The server sends the response back through the same path – Internet → ISP → Router → Your Device.

9. You See the Result:  
 Your device puts the data back together, and you see the webpage or message.

Diagram

You → Router → ISP → Internet → Server → Internet → ISP → Router → You

5) THEORY EXERCISE: Describe the roles of the client and server in web communication. Network Layers on Client and Server

A server (which provides resources or services in response). The client sends a request to the server, and the server processes the request and sends the appropriate response.

The clientis the device or software (like a browser) that makes requests for information from the server.

The serveris the system that processes requests from the client and sends back the requested data (such as web pages.

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

When we talk about a "Client," it refers to a device (usually a computer, smartphone, or application) that requests and receives services from a server. The client is the entity that initiates communication, asking for data or resources from the server. For instance, web browsers like Google Chrome, Mozilla Firefox, or Safari are common client applications that request data from a server to render web pages.

Server

A Server, on the other hand, is a  remote computer or system that provides data, resources, or services to clients. It listens to incoming client requests, processes them, and sends the required information back. A server can handle multiple client requests simultaneously.

For example, Web servers host websites, and database servers store and serve databases for applications. In simple terms, the client sends a request to the server, and the server serves the request as long as the data or service is available in its system.

REQUEST

CLINTT

PHNE-PT

SERVER

NETWORK

PC-PT

LAPTOP-PT

RESPONCE

7) THEORY EXERCISE: Explain the function of the TCP/IP model and its layers. Client and Servers

The TCP/IP model is a set of protocols used for communication over the internet. It has four layers:

1**.** Application Layer: Deals with network services like HTTP, FTP, etc.

2**.**Transport Layer: Manages end-to-end communication and error handling (TCP, UDP).

3**.** Internet Layer: Handles logical addressing and routing

(IP).

4**.** Network Interface Layer: Deals with physical transmission of data over hardware.

8) THEORY EXERCISE:

1. Explain Client Server Communication

Client-server communication is how two computers talk to each other — one is called the client (like your web browser), and the other is the server (where websites or data are stored).

How it Works (Simple Steps):

1. Client Sends a Request  
   The client asks for something — like when you open Google.com in your browser.
2. Server Receives the Request  
   The server gets the request and finds the right data (like the Google homepage).
3. Server Sends a Response  
   The server sends that data (the website) back to the client.
4. Client Displays the Response  
   Your browser receives the website data and shows it on your screen.

2 . Types of Internet Connections

1. Digital subscriber line(DSL)

2. Cable Internet

3. Fiber Optic

4. Satellite Internet

5. Wireless

6. Broadband over Power lines(BPL)

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

1) Broadband (DSL - Digital Subscriber Line)

| **Pros** | **Cons** |
| --- | --- |
| Widely available | Slower compared to cable and fiber |
| Can use phone and internet together | Speed depends on distance from provider |
| Affordable | May have limited upload speeds |

2) Cable Internet

| **Pros** | **Cons** |
| --- | --- |
| Faster than DSL | Speed can drop during peak hours |
| Widely available in cities | Shared connection with neighbors |
|  |  |

3. Fiber Optic

| **Pros** | **Cons** |
| --- | --- |
| Very fast speeds (up to 1 Gbps) | Expensive to install |
| Reliable and consistent | Not available in many rural areas |
| Best for heavy use (streaming, gaming) | Requires special infrastructure |

4. Mobile Internet

| **Pros** | **Cons** |
| --- | --- |
| Portable and convenient | Data limits or high costs on some plans |
| Useful in remote or traveling situations | Speed depends on signal strength |
| 5G offers very high speed | Battery drain on devices |

5. Satellite Internet

| **Pros** | **Cons** |
| --- | --- |
| Available in remote/rural locations | High latency (delay in data transfer) |
| No need for phone or cable lines | Slower than fiber or cable |
| Useful where other types aren't available | Weather can affect the signal |

6. Leased Line

| **Pros** | **Cons** |
| --- | --- |
| High speed and secure | Very expensive |
| Consistent performance | Not for home users |

10) THEORY EXERCISE : How does broadband differ from fiber-optic internet? Protocols

Broadbandis a high-speed internet connection that can be delivered via various technologies, such as DSL, cable, or satellite.

Fiber-optic internetuses fiber-optic cables to transmit data at very high speeds and with greater reliability compared to traditional broadband technologies.

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

Simulate HTTP Requests using curl

curl <http://example.com>

curl -H "Authorization: Bearer your\_token" <http://example.com/api/data>

curl -O <http://example.com/file.pdf>

curl -H "Authorization: Bearer your\_token" <http://example.com/api/data>

Simulate FTP Requests using curl

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

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

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

12) THEORY EXERCISE:

1. What are the differences between HTTP and HTTPS protocols ?

HTTP (Hypertext Transfer Protocol) is an unsecured protocol used for transferring web pages.

HTTPS (Hypertext Transfer Protocol Secure)is a secure version of HTTP, which uses encryption (SSL/TLS) to protect data during transmission between the client and server.

2. Application Security

Application Security refers to the measures and best practices used to protect software applications from security threats throughout their development lifecycle and during runtime .

13) LAB EXERCISE: Identify and explain three common application security vulnerabilities. Suggestpossible solutions .

### 1. Injection Attacks (e.g., SQL Injection)

* Description:

Injection vulnerabilities occur when untrusted input is sent to an interpreter (like a database), allowing attackers to execute malicious commands or queries. For example, an attacker could bypass authentication by injecting malicious SQL into a login form.

* Solution:
  + Use parameterized queries or prepared statements to prevent malicious input from being interpreted as code.
  + Validate and sanitize all user inputs rigorously.
  + Employ web application firewalls (WAFs) to block suspicious requests.

### 2. Cross-Site Scripting (XSS)

* Description:

XSS vulnerabilities occur when an application allows the injection of malicious scripts into web pages viewed by other users. These scripts can steal sensitive information like cookies or redirect users to malicious sites.

* Solution:
  + Encode output to ensure that any user-provided data displayed on a web page is treated as text and not executable code.
  + Implement a strong Content Security Policy (CSP) to restrict which scripts can run on your website.
  + Validate and sanitize user input to ensure it contains only expected data.

### 3. Broken Authentication and Session Management

* Description: Weak authentication mechanisms can allow attackers to impersonate other users or hijack sessions, often exploiting poor password policies or insecure session tokens.
* Solution:
  + Use strong, industry-standard authentication methods (e.g., OAuth 2.0, OpenID Connect).
  + Enforce multi-factor authentication (MFA) to add an extra layer of security.
  + Secure session cookies by marking them as Secure and HttpOnly, and implement mechanisms to prevent session fixation attacks.

14) THEORY EXERCISE:

1. What is the role of encryption in securing applications?

Encryption plays a key role in securing applications by converting sensitive data (like passwords, messages, or personal information) into unreadable code. This ensures that even if the data is intercepted, it cannot be understood or misused by unauthorized users.

2. Software Applications and Its Types

Software applications are programs designed to perform specific tasks for users, such as writing documents, browsing the internet, or managing data.

15) LAB EXERCISE:

Identify and classify 5 applications you use daily as either system software orapplication software

* System Software:
  + 1. Operating System (e.g., Windows, Linux, macOS).
* Application Software:
  + 1. Web Browser (e.g., Chrome, Firefox).
    2. Word Processor (e.g., Microsoft Word, LibreOffice Writer).
    3. Messaging App (e.g., WhatsApp, Slack).
    4. Media Player (e.g., VLC Player).

16) THEORY EXERCISE:

1. What is the difference between system software and application software ?

| **Feature** | **System Software** | **Application Software** |
| --- | --- | --- |
| **Purpose** | Runs and manages computer hardware and system | Performs specific tasks for the user |
| **Examples** | Windows, Linux, macOS, Drivers | MS Word, Excel, Chrome, WhatsApp |

2) Software Architecture

Software Architecture is the basic structure of a software system. It shows how different parts of the software are organized and how they work together. It includes decisions about design, components, and how data flows.

In short ,

Software architecture is the blueprint of a software system that helps developers build, manage, and improve it.

17)LAB EXERCISE:

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

| Presentation Layer |

| (Web Browser / UI) |

|

|

| Application Layer |

| (Server / Business |

| Logic) |

|

v

-------------------------------------

| Data Layer |

| (Database Server) |

Explanation of Layers:

1. Presentation Layer:
   * Interface seen by users (HTML, CSS, JS).
   * Example: Web browser or mobile app.
2. Application Layer:
   * Handles business logic and processing.
   * Example: PHP, Java, Node.js server.
3. Data Layer:
   * Stores and manages data.
   * Example: MySQL, MongoDB.

18) THEORY EXERCISE:

1 )What is the significance of modularity in software architecture? Layers in Software Architecture

Modularity means breaking the software into small, separate parts (called modules), each doing a specific task.

Explanation of Layers:

1. Presentation Layer:
   * Interface seen by users (HTML, CSS, JS).
   * Example: Web browser or mobile app.
2. Application Layer:
   * Handles business logic and processing.
   * Example: PHP, Java, Node.js server.
3. Data Layer:
   * Stores and manages data.
   * Example: MySQL, MongoDB.

19) LAB EXERCISE:

1) Create a case study on the functionality of the presentation, business logic, and dataaccess layers of a given software system.

Example: For an e-commerce system:

* Presentation Layer: Displays product catalogs, search options, and user profiles.
* Business Logic Layer: Handles cart management, checkout processes, and discounts.
* Data Access Layer: Retrieves product information, processes payments, and updates user data.

20)THEORY EXERCISE:

1.Why are layers important in software architecture?

Layers are important because they help organize the software in a clean and structured way. Each layer has a specific role, making the system easier to develop, understand, test, and maintain.

2.software environment

A software environment is the setup in which software is developed, tested, and run. It includes the hardware, operating system, tools, and software needed for a program to work properly.

20) LAB EXERCISE:

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

1) Development Environment – Where developers write and build the software.  
*Example: Visual Studio, Eclipse.*

2) Testing Environment – Used to test the software before release.  
*Example: QA tools, test servers.*

3) Staging Environment – A copy of the real system used for final testing.  
*Example: Pre-production setup.*

4) Production Environment – The live system where users use the software.  
*Example: Website or app running for public users.*

21)THEORY EXERCISE:

1. Explain the importance of a development environment in software production.

A development environment provides the tools, libraries, and frameworks required to write, test, and debug software. It ensures consistency and efficiency during the development process, enabling developers to build and test applications in a controlled environment before deploying them.

2. Source Code

Source code is the original set of instructions written by programmer using a programming language like C, C++, Java, Python, or PHP.

It tells the computer what to do, and it's written in a human-readable form before being converted into machine code.

22)LAB EXERCISE:

1. Write and upload your firstsource code file to Github

Follwing are the command to uploaad file

git config --global user.name "Sachin"

git config --global user.email "mahajansachin@example.com"

git init

git add hello.doc

git commit -m "first source code "

git remote add origin https://github.com/yourusername/my-first-code.git

git push -u origin master

23) THEORY EXERCISE:

1. What is the difference between source code and machine code?

Source codeis the human-readable code written by developerin high-level programming languages.

Machine codeis the low-level, binary code that the computer's processor can directly execute

24) LAB EXERCISE:

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

git config --global user.name "Sachin"

git config --global user.email "mahajansachin@example.com"

git init

git add hello.doc

git commit -m "first source code "

git remote add origin https://github.com/yourusername/my-first-code.git

git push -u origin master

24) THEORY EXERCISE: Why is version control important in software development?

Version control allows developers to track changes to the source code over time, collaborate with others, and manage different versions of the software. It ensures that changes can be reversed if necessary and provides a clear history of modifications.

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

* Create a student account on GitHub:
  1. Visit [GitHub Student Developer Pack](https://education.github.com/pack).
  2. Sign up using your student credentials.
  3. Collaborate on a small project with a classmate by forking a repository or creating branches.

26) LAB EXERCISE: Create a list of software you use regularly and classify them into the followingcategories: system, application, and utility software.

* + System Software:
    1. Operating System.
  + Application Software:
    1. Web Browsers (Chrome).
    2. Messaging Tools (WhatsApp).
  + Utility Software:
    1. Antivirus Software.
    2. Disk Cleanup Tools.

27) THEORY EXERCISE: What are the differences between open-source and proprietary software?

Open-source software is a type of software that anyone can use, see**,** change, and share. The source code (the behind-the-scenes programming) is open to everyone. People can even improve it or fix problems.

Example - Linux, VLC Media Player, Mozilla Firefox

Proprietary software **is owned by a company or person**, and you cannotsee or change the source code. You have to buy or get a license to use it, and you can’t copy or modify it without permission.

Example - Microsoft Windows, Adobe Photoshop.

28) THEORY EXERCISE: How does GIT improve collaboration in a software development team?

GIT helps a software team work together by allowing multiple people to work on the same project at the same time without losing any work. It keeps track of **all changes**, so if someone makes a mistake, it can be easily fixed. Team members can work on different features separately and then combine their work safely. This makes teamwork faster, organized, and safer.

29 ) LAB EXERCISE: Write a report on the various types of application software and how they improve productivity.

Application software is a type of computer program designed to help users perform specific tasks.

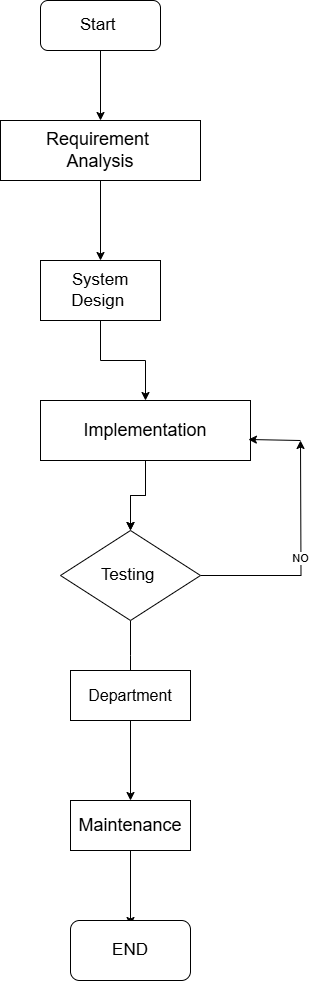
Types**:**

* Word Processors (e.g., LibreOffice Writer): Enhance documentation and report creation.
* Spreadsheets (e.g., Excel): Aid in organizing and analyzing data.
* Presentation Software (e.g., PowerPoint): Improve communication through visual storytelling.
* Communication Tools (e.g., Zoom): Facilitate remote collaboration.
* Database Management Systems (e.g., MySQL): Streamline data storage and retrieval.

30) THEORY EXERCISE: What is the role of application software in businesses?

* Application software helps businesses perform their daily tasks more efficiently and accurately.
* It is used for things like managing data, creating documents, sending emails, making presentations, accounting, and customer management.
* It improves productivity, saves time, reduces errors, and helps in better decision-making.
* With the help of software like MS Excel, Tally, and CRM tools, businesses can grow faster and work smarter.

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



32) THEORY EXERCISE: What are the main stages of the software development process?

The main stages of the software development process are:

1. Planning – Deciding what the software should do and setting goals.
2. Requirement Analysis – Understanding what the users need.
3. Design – Creating the structure and layout of the software.
4. Development – Writing the code and building the software.
5. Testing – Checking the software for errors and bugs.
6. Deployment – Releasing the software to users.
7. Maintenance – Fixing problems and making updates after release.

33)LAB EXERCISE: Write a requirement specification for a simple library management system.

Funtional Requiremnet

1) User Login System

* Admin and members can log in using a username and password.

2) Manage Books

* Add, update, delete, and search book details (title, author, ISBN, quantity).

3) Manage Members

* Register new members and store their personal details (name, email, phone number).

4) Issue Book

* Allow members to borrow books if available.
* Record issue date and return date.

5) Return Book

* Mark books as returned.
* Calculate and display any late fines.

5) Search Function

* Search for books by title, author, or ISBN.

6) Reports

* Generate reports of issued books, available books, and defaulters (late returns).

2. Non-Functional Requirements

* Usability: The system should be easy to use.
* Security: Only authorized users can access or update data.
* Performance: The system should work quickly and handle many records.

**3 .** System Requirements

* Frontend: HTML/CSS
* Backend: PHP / Java / Python (any one)
* Database: MySQL or SQLite
* Platform: Web-based or Desktop

34) THEORY EXERCISE: Why is the requirement analysis phase critical in software development?  
 The requirement analysis phase is critical because it helps understand exactly what the users need from the software.

If this step is done well, developers can build the right software that meets user expectations. It prevents mistakes, reduces costs, and saves time by avoiding changes later.

Without proper requirement analysis, the software may not work correctly or satisfy users.

35) LAB EXERCISE: Perform a functional analysis for an online shopping system.

A. Customer Functions

1. Registration & Login
   * Register with name, email, password
   * Login with credentials
   * Forgot password/reset option
2. Product Browsing and Search
   * View product listings
   * Filter by category, price, rating
   * Search products by name or keyword
3. Product Details
   * View detailed information (description, price, reviews)
   * View product images
4. Shopping Cart
   * Add/remove products
   * Update quantity
   * View cart total
5. Order Placement
   * Checkout process with address input
   * Select payment method
   * Place order
6. Payment
   * Payment gateway integration (UPI, credit card, net banking)
   * Secure transaction processing
7. Order Tracking
   * View current order status (Pending, Shipped, Delivered)
   * Estimated delivery date
8. Order History
   * View previous orders and invoices
9. Profile Management
   * Update personal details (name, address, phone)
   * Change password
10. Product Reviews & Ratings

* Leave a rating and comment on purchased products

B. Admin Functions

1. Admin Login
   * Secure admin login with fixed credentials
2. User Management
   * View list of registered customers
   * Block or delete user accounts
3. Product Management
   * Add new products (name, image, description, price, category)
   * Edit or delete products
4. Order Management
   * View all customer orders
   * Update order status
5. Inventory Management
   * Monitor product stock levels
   * Set restock alerts
6. Report Generation
   * Generate sales reports
   * View top-selling products

**1.** Non-Functional Requirements

* Security: SSL, secure login, encrypted payments
* Usability: Clean UI/UX for easy navigation
* Scalability: Handle large product catalogs and users
* Performance: Fast page load and processing times
* Reliability: Minimal downtime, robust database

36) THEORY EXERCISE: What is the role ofsoftware analysis in the development process?

Software analysis helps to clearly understand what the software should do before building it. It involves gathering and studying user needs, defining system requirements, and creating a plan for developers to follow.

* Understands what the user wants
* Defines clear requirements
* Helps in planning the project
* Reduces errors and misunderstandings
* Acts as a guide for design, development, and testing

37) LAB EXERCISE: Design a basic system architecture for a food delivery app.

Components:

Frontend (User Interface)**:** Displays menu, location, and order tracking.

Backend (Business Logic)**:** Processes orders, calculates delivery times.

Database: Stores user data, menu details, and order history.

38) THEORY EXERCISE: What are the key elements ofsystem design?

1. Architecture Design  
   – Decides how the system is built (frontend, backend, database).
2. User Interface (UI) Design  
   – How the app looks and works for users.
3. Data Design  
   – How data is stored and managed in the system.
4. Module Design  
   – Divides the system into small parts (modules) to make it easier to build and maintain.
5. Security Design  
   – Protects the system from hackers and unauthorized access.
6. Scalability  
   – Makes sure the system can grow (handle more users/data) smoothly.
7. Reliability  
   – Ensures the system keeps working even if something goes wrong.
8. Integration  
   – Connects the system with other services (like payment, maps, etc).

39) LAB EXERCISE: Develop test cases for a simple calculator program.

Addition**:**

* + - Input: 2 + 3.
    - Expected Output: 5.

Division**:**

* + - Input: 10 ÷ 2.
    - Expected Output: 5.

Sbstration

* + - Input: 30 – 10
    - Expected Output : 20

Expand for other operations like multiplication, and handling edge cases (e.g., dividing by zero).

(e.g., dividing by zero).

40) THEORY EXERCISE: Why is software testing important?

Software testing is important because it helps to:

1. Find and fix bugs early
2. Ensure the software works correctly
3. Improve quality and performance
4. Prevent future errors
5. Increase user trust and satisfaction

41) LAB EXERCISE: Document a real-world case where a software application required criticalmaintenance.

Case: Online Banking App Bug Fix

* What happened**:**  
  An online banking app had a bug that caused incorrect display of account balances for many users.

Why maintenance was needed**:**

* The bug created confusion and risk of wrong transactions.
* The bank had to quickly fix the software to show correct balances.

Outcome:

* Developers performed critical maintenance to fix the bug.
* The app was updated and tested before going live again.
* Users’ trust was restored.

42) THEORY EXERCISE: What types ofsoftware maintenance are there?

1. Corrective Maintenance  
   – Fixing bugs and errors in the software.
2. Adaptive Maintenance  
   – Updating software to work with new hardware or operating systems.
3. Perfective Maintenance  
   – Improving performance or adding new features.
4. Preventive Maintenance  
   – Making changes to prevent future problems.

43) THEORY EXERCISE: What are the key differences between web and desktop applications?

Web apps run in browsers and need internet, while desktop apps run on your computer and work offline

| Feature | Web Application | Desktop Application |
| --- | --- | --- |
| Installation | No need to install, runs in browser | Must be installed on the computer |
| Internet Required | Needs internet to work | Works offline (mostly) |
| Access | Can be used from any device with browser | Only works on the device it is installed on |
| Updates | Easy to update from the server | Needs manual updates on each device |
| Performance | Slower (depends on internet) | Usually faster (runs on local system) |

44) THEORY EXERCISE: What are the advantages of using web applications over desktop applications?

1. **No Installation Needed**  
– > Runs directly in the browser.

2. **Access Anywhere**  
– > Can be used from any device with internet.

3. **Easy to Update**  
–> Updates are done on the server side, no need to reinstall.

4. **Cross-Platform**  
–> Works on Windows, Mac, Linux, etc.

5. **Lower Maintenance Cost**  
– > Easier to manage and support users.

45) THEORY EXERCISE: What role does UI/UX design play in application development?

UI/UX design plays a key role in application development by making the app:

* Easy to use (UI – User Interface)
* Enjoyable and smooth to **experience** (UX – User Experience)

It helps users understand and interact with the app better, improving satisfaction and making the app more successful.

46) THEORY EXERCISE: What are the differences between native and hybrid mobile apps?

Native apps are built specifically for one platform like Android or iOS, so they run faster and provide a better user experience. They can fully access device features like the camera and GPS but take more time and cost to develop.

Hybrid apps, on the other hand, are built using one codebase for multiple platforms. They are quicker and cheaper to develop but may have slightly lower performance and limited access to device features.

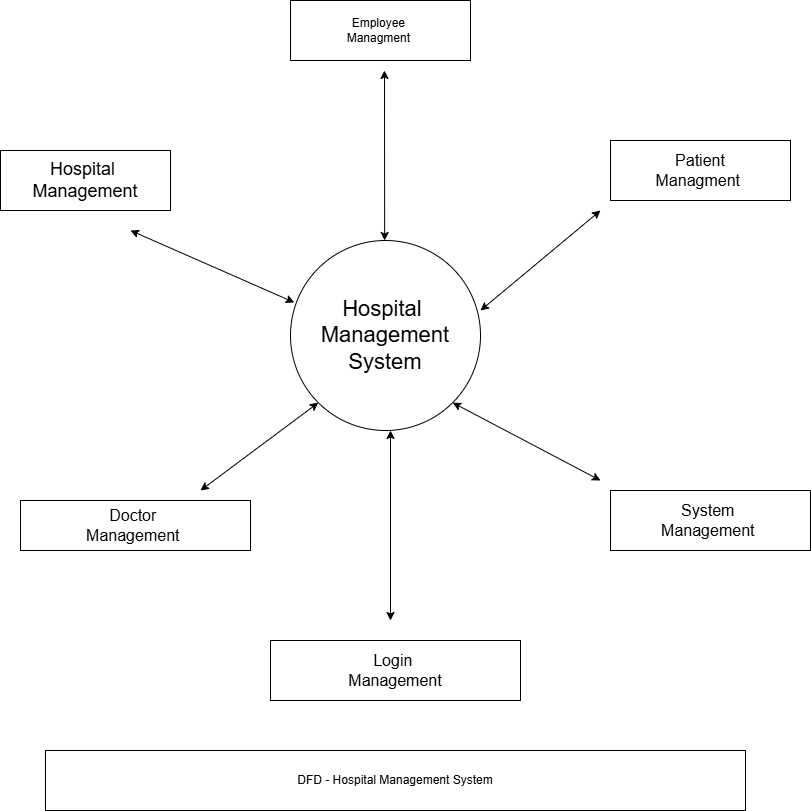
47 )THEORY EXERCISE: What is the significance of DFDs in system analysis?.

DFDs play a vital role in system analysis as they visually represent the flow of data within a system.

They help analysts, developers, and stakeholders understand how data is processed, where it comes from, and where it goes.

By breaking down complex systems into simpler parts, DFDs make it easier to identify errors, improve system design, and ensure that all functional requirements are met. They are useful tools for planning and communication during the development process.

48) LAB EXERCISE: Create a DFD for a hospital management system.



49) LAB EXERCISE: Build a simple desktop calculator application using a GUI library

Use a language like Python and the Tkinter library to create a calculator that can perform basic operations like addition, subtraction, multiplication, and division.

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

**Desktop Applications**

**Pros:**

* Work offline
* Faster performance
* Better access to system resources (files, hardware)

**Cons:**

* Need to be installed
* Harder to update (manual updates)
* Tied to specific OS (Windows, Mac, etc.)

**Web Applications**

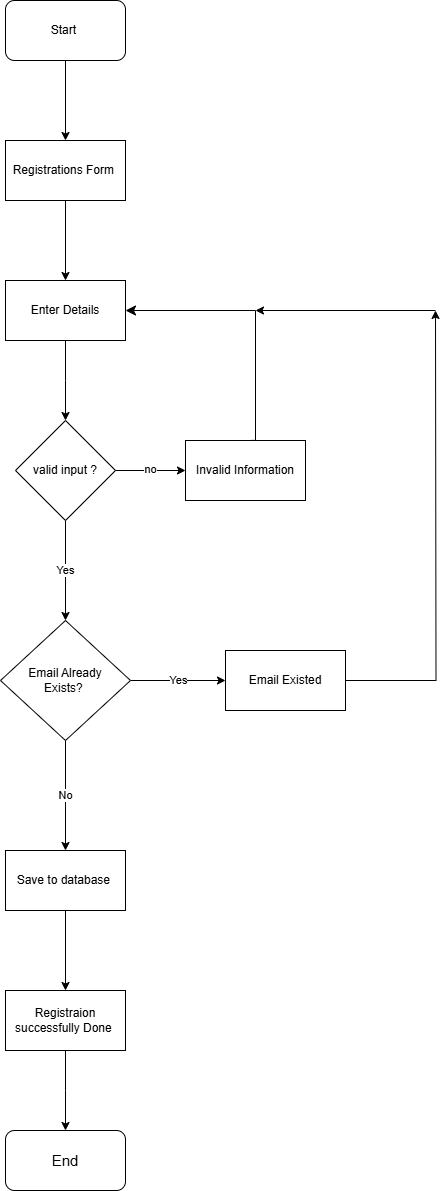
**Pros:**

* No installation needed (run in browser)
* Easy to update (server-side)
* Work on any device with a browser

**Cons:**

* Need internet connection
* Can be slower
* Limited access to local files/hardware

51) LAB EXERCISE: Draw a flowchart representing the logic of a basic online r egistration system.



Flowchart of Online Registration System

52) THEORY EXERCISE: How do flowcharts help in programming and system design ?

Flowcharts are important tools that help programmers and designers by showing the step-by-step process of how a program or system works.

They use symbols and arrows to represent decisions, actions, and flow of control. This visual representation makes it easier to understand the logic, plan the program before writing code, and identify mistakes early.

Flowcharts also help communicate ideas clearly among team members or users, making the development process faster and more efficient.