Module 1: Labwork

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

**C Language**

C is a procedural programming language. It was initially developed by Dennis Ritchie as a system programming language to write an operating system.

**Features:**

* Include low-level access to memory, a simple set of keywords, and a clean style, these features make C language suitable for system programming like operating system or compiler development.
* Fast and Efficient.

**Hello World Program in C Language:**

#include <stdio.h>

​

int main() {

   printf("Hello World");

   return 0;

}

**. C++ Language**

C++, high-level computer programming language. Developed by Bjarne Stroustrup of Bell Laboratories in the early 1980s, it is based on the traditional C language but with added object-oriented programming and other capabilities.

**Features:**

* Machine Independent or Portable.
* Mid-level programming language.

**Hello World Program in C++:**

#include <iostream>

​

int main() {

    std::cout << "Hello World";

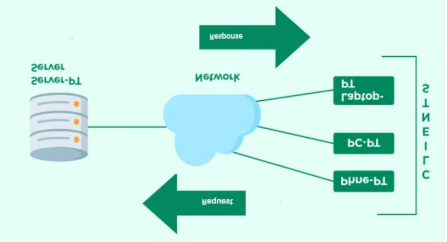
    return 0;

}

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

It follows the request response pattern using networking protocols like TCP/IP and HTTP/HTTPS.

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



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

. Fiber Optic:

**Pros:** Fastest speeds, symmetrical upload and download speeds, very reliable, less susceptible to interference.

* **Cons:** Limited availability in some areas, potentially higher cost than other options.

2. Cable:

* **Pros:** Generally fast speeds, widely available, reliable.
* **Cons:** Speeds can slow down during peak hours due to shared bandwidth, can be less reliable than fibre.

3. DSL (Digital Subscriber Line):

* **Pros:** Relatively affordable, widely available, good for basic internet use.
* **Cons:** Slower speeds compared to fibre and cable; speeds can decrease with distance from the telephone exchange.

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

Curl is a command line tool for doing all sorts of URL manipulations and transfers, but this particular document focuses on how to use it when doing HTTP requests for fun and profit. This document assumes that you know how to invoke curl --help or curl --manual to get basic information about it.  It makes the requests, it gets the data, it sends data and it retrieves the information. You probably need to glue everything together using some kind of script language or repeated manual invokes.

6.  Identify and explain three common application security vulnerabilities.

Three common application security vulnerabilities are SQL Injection, Cross-Site Scripting (XSS), and Broken Authentication. SQL injection allows attackers to execute malicious SQL code, potentially gaining access to sensitive data. XSS enables attackers to inject malicious scripts into web pages viewed by other users. Broken authentication allows attackers to bypass login mechanisms and gain unauthorized access.

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

1. **Operating System (OS):**

The OS (like Windows, macOS, or Linux) is the core system software that manages all other programs and hardware resources. It provides a platform for applications to run and interacts with the computer's hardware.

1. **Device Drivers:**

These software components allow the operating system to communicate with specific hardware devices (e.g., printers, graphics cards, network cards). Without drivers, the OS wouldn't be able to use these devices.

1. **Firmware:**

Firmware is a type of software embedded in hardware devices (like a router or motherboard) that controls the device's basic functions. It's essentially the software that makes the hardware work.

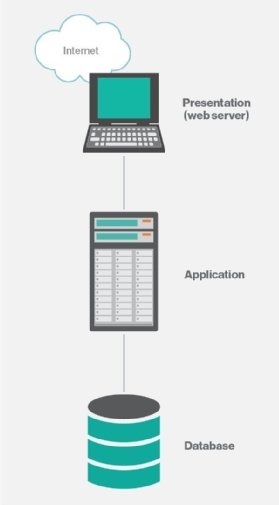
1. **Utility Software:**

These tools are used for system maintenance and optimization. Examples include antivirus software, disk defragmenters, and disk repair tools. They help keep the computer running smoothly.

1. **BIOS/UEFI:**

These are low-level software programs that initialize the computer hardware when it starts up. BIOS (Basic Input/Output System) is older, while UEFI (Unified Extensible Firmware Interface) is the newer standard.

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



Three-Tier-Client-Server-Architecture-in-Distributed-SystemThree-Tier-Client-Server-Architecture-in-Distributed-System

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

1.  Presentation Layer (UI Layer):

* **Function:**

This layer is the user's point of contact with the application. It displays information to the user, gathers user input, and communicates with the business logic layer to process requests.

2.  Business Logic Layer (Application Layer):

* **Function:**

This layer contains the core logic and rules of the application. It processes data, manages workflows, and enforces business rules. It acts as an intermediary between the presentation layer and the data access layer.

3.  Data Access Layer (DAL):

* **Function:**

This layer provides an abstraction layer for interacting with the database. It hides the complexities of database interactions from the business logic layer.

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

* **Development Environment:**
* **Purpose:** A flexible space for developers to write, debug, and experiment with code.
* **Characteristics:** Typically has fewer restrictions, allowing for rapid iteration and customization. Developers use IDEs, code editors, and version control systems.
* **Testing Environment:**
* **Purpose:** Dedicated to rigorous testing of the software to identify and resolve bugs.
* **Characteristics:** May include various testing types like unit, integration, functional, performance, and security tests.
* **Production Environment:**
* **Purpose:** The live environment where end-users access and interact with the software.
* **Characteristics:** Requires stability, security, and reliability, with strict change management processes.

Setting up a Development Environment in a Virtual Machine:

1. **Install a Virtualization Platform:** Download and install virtualization software like VirtualBox or VMware.
2. **Create a New Virtual Machine:** Within the virtualization software, create a new virtual machine, specifying the desired operating system and resources (CPU, RAM, disk space).
3. **Install the Operating System:** Install the chosen operating system (e.g., Ubuntu, Windows) on the virtual machine, following the prompts.
4. **Install Development Tools:** Install the necessary tools for your project.

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

#include<stdio.h>

Int main(){

Printf(“My name is Dhruvan”);

}

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

**Initializing a Git repository**

1. Open Terminal .
2. Navigate to the root directory of your project.
3. Initialize the local directory as a Git repository. By default, the initial branch is called main . ...
4. Add the files in your new local repository. ...
5. Commit the files that you've staged in your local repository.

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

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

System Software:

* **Operating System:** Windows 11 (or your preferred OS like macOS, Linux, etc.). This manages the computer's resources and provides a platform for other software.
* **Device Drivers:** These allow the operating system to communicate with hardware components (e.g., graphics card driver, printer driver).

Application Software:

* **Web Browser:** Chrome, Firefox, or Safari (for browsing the internet).
* **Productivity Suite:** Microsoft Office (Word, Excel, PowerPoint, etc.) or Google Workspace (Docs, Sheets, Slides, etc.) for document creation, spreadsheets, and presentations.
* **Email Client:** Outlook, Gmail (for managing emails).
* **Media Player:** VLC, Windows Media Player (for playing audio and video file).

Utility Software:

* **Antivirus Software:** McAfee, Norton, or Windows Defender (for protecting against malware).
* **Compression Tool:** WinRAR, 7-Zip (for compressing and decompressing files).
* **Disk Management Tools:** Disk Cleanup, Disk Defragmenter (for optimizing disk space and performance).

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

Git is a popular version control system that allows developers to track changes in their code and collaborate with others. Cloning a branch in Git involves creating a copy of a specific branch from a remote repository.  Branch in Git is simply a lightweight movable pointer to one of these commits. Merging in Git is the process of combining the independent lines of development from one branch into another.

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

* **Productivity Software:**

This category includes tools like word processors (e.g., Microsoft Word, Google Docs), spreadsheets (e.g., Microsoft Excel, Google Sheets), and presentation software (e.g., Microsoft PowerPoint, Google Slides).

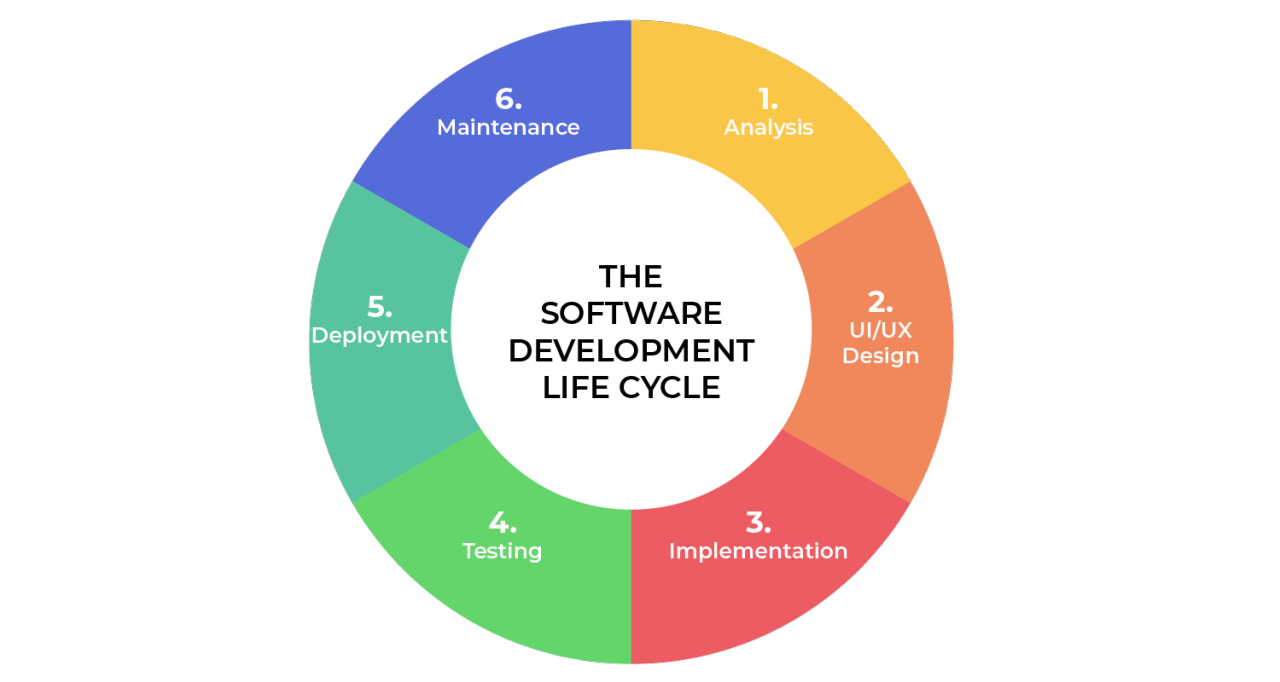
* **Impact:** Streamlines document creation, data analysis, and presentation design, saving time and improving the quality of deliverables.
* **Communication and Collaboration Software:** Platforms like Slack, Microsoft Teams, and email clients facilitate communication and collaboration among teams, enabling real-time updates, file sharing, and project coordination.
* **Impact:** Improves team communication, reduces email clutter, and accelerates decision-making processes.
* **Project Management Software:**

Tools like Asana, Trello, and Jira help in planning, organizing, and tracking projects, ensuring tasks are completed on time and within budget.

* **Impact:** Enhances project visibility, reduces bottlenecks, and improves resource allocation.
* **Customer Relationship Management (CRM) Software:**

CRM systems like Salesforce and HubSpot manage customer interactions, track sales pipelines, and improve customer service.

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



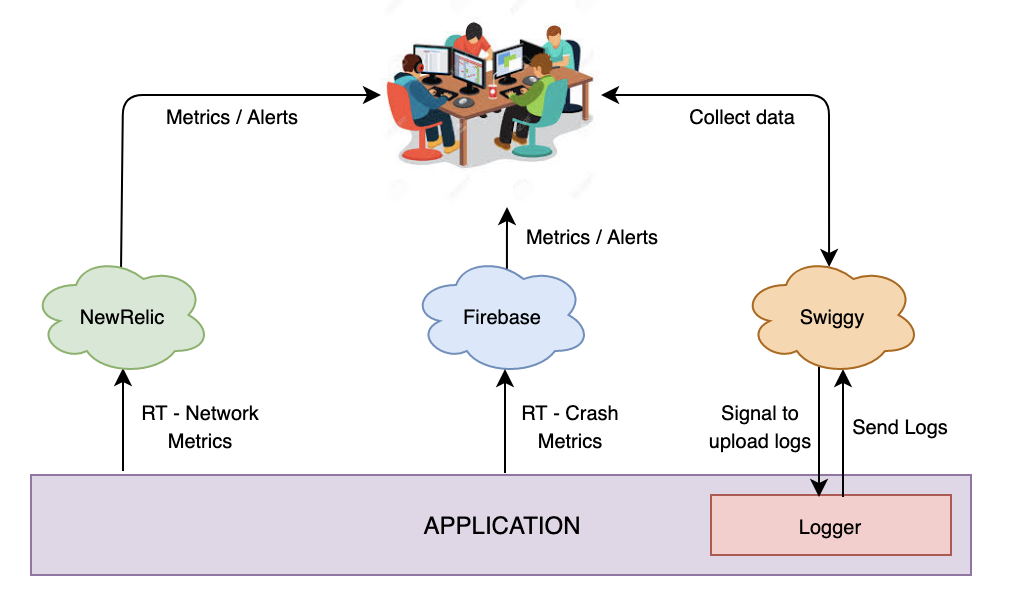
18.  Write a requirement specification for a simple library management system.

The system should be user-friendly so that it is easy to use for the users. The information of all users, books and libraries must be stored in a database that is accessible by the website. The system should have more storage capacity and provide fast access to the database.

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

A functional analysis of an online shopping system identifies the core features and behaviors the system must provide. This includes functionalities like product browsing, searching, adding to cart, checkout, payment processing, order management, user account management, and customer service features. It also covers non-functional aspects like security, performance, and usability.

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



21.  Develop test cases for a simple calculator program.

 Add one or more numbers and verify the calculated result when you hit =.

●      Subtract one or more numbers and verify the calculated result when you hit =.

●      Multiply one or more numbers and verify the calculated result when you hit =.

●      Divide a few numbers and verify the calculated result when you hit =.

●      Verify the AC and C buttons clear the result screen when you hit =.

●      Add, subtract, and multiply using a negative number when you hit =.

●      Add, subtract, multiply, and divide by numbers with decimals when you hit =.

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

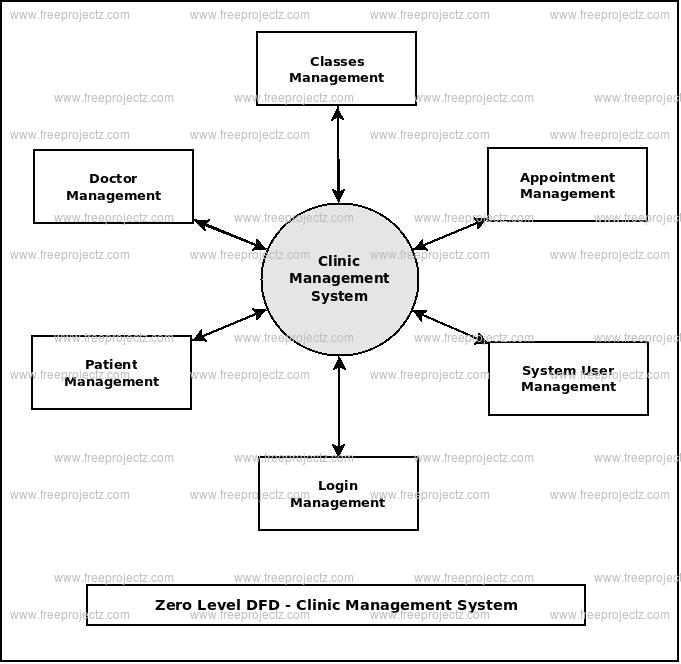
1. A real-world example of critical software maintenance involved a large, publicly traded bank's online banking platform.

2. The bank's software maintenance team was immediately mobilized to address the issue. The first step was to isolate the affected system to prevent further unauthorized access. This involved temporarily taking the online banking platform offline. Then, the team launched an investigation into the vulnerability, identifying the root cause and the extent of the damage.

3. The maintenance team then implemented a patch to fix the identified vulnerability. This involved modifying the authentication code and implementing additional security measures to prevent similar vulnerabilities in the future.

4. After implementing the patch, the team thoroughly tested the platform to ensure that the vulnerability was resolved and that no other issues were introduced. This included both functional and security testing to verify that the system was functioning correctly and that the new security measures were effective.

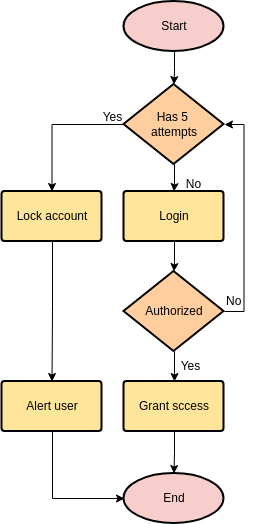
23.  Create a DFD for a hospital management system.



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

DOUBT

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



online registration module flowchartonline registration module flowchartonline registration module flowchartonline registration module flowchart

Three-Tier-Client-Server-Architecture-in-Distributed-System